Proceedings of the 15th ISWC workshop on Ontology Matching (OM 2020)

Pavel Shvaiko, Jérôme Euzenat, Ernesto Jiménez-Ruiz, Oktie Hassanzadeh, Cassia Trojahn dos Santos

Ontology Matching

OM-2020

Proceedings of the ISWC Workshop

Introduction

Ontology matching[1] is a key interoperability enabler for the semantic web, as well as a useful tactic in some classical data integration tasks dealing with the semantic heterogeneity problem. It takes ontologies as input and determines as output an alignment, that is, a set of correspondences between the semantically related entities of those ontologies. These correspondences can be used for various tasks, such as ontology merging, data translation, query answering or navigation over knowledge graphs. Thus, matching ontologies enables the knowledge and data expressed with the matched ontologies to interoperate.

The workshop had three goals:

- To bring together leaders from *academia*, *industry* and *user institutions* to assess how academic advances are addressing real-world requirements. The workshop strives to improve academic awareness of industrial and final user needs, and therefore, direct research towards those needs. Simultaneously, the workshop serves to inform industry and user representatives about existing research efforts that may meet their requirements. The workshop also investigated how the ontology matching technology is going to evolve.
- To conduct an extensive and rigorous evaluation of ontology matching and instance matching (link discovery) approaches through the OAEI (Ontology Alignment Evaluation Initiative) 2020 campaign[2].
- To examine similarities and differences from other, old, new and emerging, techniques and usages, such as process matching, web table matching or knowledge embeddings.

The program committee selected 6 long and 4 short submissions for oral presentation and 6 submissions for poster presentation. 19 matching systems participated in this year's OAEI campaign. Further information about the Ontology Matching workshop can be found at: `http://om2020.ontologymatching.org/`.

[1] `http://www.ontologymatching.org/`
[2] `http://oaei.ontologymatching.org/2020`

Acknowledgments. We thank all members of the program committee, authors and local organizers for their efforts. We appreciate support from the Trentino as a Lab[3] initiative of the European Network of the Living Labs[4] at Trentino Digitale[5], the EU SEALS (Semantic Evaluation at Large Scale) project[6], the EU HOBBIT (Holistic Benchmarking of Big Linked Data) project[7], the Pistoia Alliance Ontologies Mapping project[8] and IBM Research[9].

Pavel Shvaiko
Jérôme Euzenat
Ernesto Jiménez-Ruiz
Oktie Hassanzadeh
Cássia Trojahn

December 2020

[3] www.facebook.com/trentinoasalab
[4] www.openlivinglabs.eu
[5] www.trentinodigitale.it
[6] www.seals-project.eu
[7] https://project-hobbit.eu/challenges/om2020/
[8] www.pistoiaalliance.org/projects/current-projects/ontologies-mapping
[9] research.ibm.com

Organization

Organizing Committee

Pavel Shvaiko,
Trentino Digitale SpA, Italy

Jérôme Euzenat,
INRIA & University Grenoble Alpes, France

Ernesto Jiménez-Ruiz,
City, University of London, UK & SIRIUS, University of Oslo, Norway

Oktie Hassanzadeh,
IBM Research, USA

Cássia Trojahn,
IRIT, France

Program Committee

Alsayed Algergawy, *Jena University, Germany*
Manuel Atencia, *University Grenoble Alpes & INRIA, France*
Zohra Bellahsene, *LIRMM, France*
Jiaoyan Chen, *University of Oxford, UK*
Valerie Cross, *Miami University, USA*
Jérôme David, *University Grenoble Alpes & INRIA, France*
Gayo Diallo, *University of Bordeaux, France*
Daniel Faria, *Instituto Gulbenkian de Ciéncia, Portugal*
Alfio Ferrara, *University of Milan, Italy*
Marko Gulić, *University of Rijeka, Croatia*
Wei Hu, *Nanjing University, China*
Ryutaro Ichise, *National Institute of Informatics, Japan*
Antoine Isaac, *Vrije Universiteit Amsterdam & Europeana, Netherlands*
Naouel Karam, *Fraunhofer, Germany*
Prodromos Kolyvakis, *EPFL, Switzerland*
Patrick Lambrix, *Linköpings Universitet, Sweden*
Oliver Lehmberg, *University of Mannheim, Germany*
Majeed Mohammadi, *TU Delft, Netherlands*
Peter Mork, *MITRE, USA*
Andriy Nikolov, *Metaphacts GmbH, Germany*
George Papadakis, *University of Athens, Greece*
Catia Pesquita, *University of Lisbon, Portugal*

Table of Contents

OAEI Papers

Posters

Using Domain Lexicon and Grammar for Ontology Matching

Francisco José Quesada Real[1,2], Gábor Bella[3], Fiona McNeill[1], and Alan Bundy[1]

[1] University of Edinburgh, UK
{s1580097,fmcneill,a.bundy}@ed.ac.uk
[2] University of Cádiz, Spain
franciscojose.quesada@uca.es
[3] University of Trento, Italy
gabor.bella@unitn.it

Abstract. There are multiple ontology matching approaches that use domain-specific background knowledge to match labels in domain ontologies or classifications. However, they tend to rely on lexical knowledge and do not consider the specificities of domain grammar. In this paper, we demonstrate the usefulness of both lexical and grammatical linguistic domain knowledge for ontology matching through examples from multiple domains. We also provide an evaluation of the impact of such knowledge on a real-world problem of matching classifications of mental illnesses from the health domain. Our experimentation with two matcher tools that use very different matching mechanisms—LogMap and SMATCH—shows that both lexical and grammatical knowledge improve matching results.

Keywords: Ontology Matching · Domain-Knowledge · Domain Language · Domain Lexicon · Domain Grammar

1 Introduction

Ontology Matching (OM) aims at finding correspondences between the classes and instances of multiple ontologies [10]. Thus, OM processes are commonly carried out to solve heterogeneity problems that occur when multiple knowledge resources need to be integrated or used together. Among common approaches used in OM, the comparison of node labels has been one of the most performant and widely used techniques. While label matching has been addressed by the earliest matchers through simple methods such as string similarity, more complex cases such as syonymy, cross-lingual, or domain-specific matching need linguistically better-founded solutions [3]. The problem of matching domain ontologies or classifications is special because labels tend to mix elements of the general language with domain terms, and sometimes even grammatical

forms that are domain-specific. In cross-domain matching scenarios, phenomena of meaning shifts, polysemy, and synonymy make the matching task even harder, such as in the emergency response domain where subdomains of police, healthcare, fire brigades, etc., need to be aligned [17, 18]. Another example is that of mapping standard classifications within the healthcare domain that, despite relying on precise domain terminology, express the same concepts in different ways, such as *'rupture of aorta'* versus *'aortic aneurysm, ruptured'*. Establishing precise mappings across standards has a major importance for cross-border health applications as they enable automated data integration methods [6].

A large number of matchers analyse natural language labels, on different levels of complexity. A common approach is to incorporate linguistic *background knowledge* (BK) into the matcher [9, 10].

SMATCH [14] relies on domain-independent BK: it uses WordNet [12] as an English domain-independent lexical database, and analyses labels using general grammatical tools such as tokeniser, lemmatiser, and syntactic parser. Other matchers, such as LogMap [15] or YAM-BIO [2], have been customised to integrate domain terminology to address specific matching challenges, such as biomedical terms. This results in increased performance on domain-specific matching; however, the longer the labels become, the more likely their grammatical structures and their use of general language become important, which cannot be covered by terminological knowledge alone. For this reason, some matchers, such as AML [11] or ALIN [8], integrate both domain-independent and domain-specific knowledge (e.g. WordNet together with biomedical resources).

All of these matchers, however, are limited to using lexical BK. While some of them do address grammar through basic domain-independent methods (tokenisation, lemmatisation, stop word elimination), they do not cope with cases where the grammar depends on the domain.

In this paper, we investigate the impact of both domain lexicon and domain grammar in ontology matching, mainly focusing on label-based matching.

The paper is organised as follows. In Section 2 we describe how domain knowledge appears in ontology labels. Section 3 focusses on different approaches that matchers may use to take advantage of domain knowledge. A case study on the health domain is presented in section 4, being evaluated in section 5. The paper finishes with some concluding remarks and future works included in section 6.

2 Domain Language in Ontology Labels

The use of specialised linguistic constructs is common in most domains of knowledge. The most obvious case is the use of specialised terminology, consisting of words and expressions that either are used exclusively within the context of a domain (such as *to deglaze* in cooking meaning *'to loosen bits of food which stuck on the bottom of a pan by adding liquid'*), or that gain a new meaning within a domain (such as *to clarify* which in cooking refers specifically to butter).

Domain-specific meaning can, however, also be vehicled by non-lexical means, a phenomenon that we globally call *domain grammar*. Domain grammar can be found even within the short labels typical of ontologies and classifications. Below we provide examples of domain language from specialised text, including ontology labels.

Domain terms. The *UK Civil & Protection Lexicon* (UKCP) defines the term *medevac* that means *medical evacuation*, itself considered a specialised term. In order to align these two terms, a matcher would either need lexical background knowledge that states their synonymy, or—in this specific case—word-level analysis in order to detect that one term is the abbreviated form of the other.

Domain acronyms. The acronym *REM* has many meanings; in the domain of neurology it means *rapid eye movement*. Again, in a matching task the acronym can be matched either through the use of domain lexical knowledge or through acronym detection.

Word derivation. Derivation rules allow the creation of words through the use of affixes, such as *voyeur* $\mapsto$ *voyeurism* or *anorexia* $\mapsto$ *anorexic*. While, as in these cases, domain language often relies on the derivational rules of general grammar, domain-specific derivational affixes and rules also exist, such as *candida* $\mapsto$ *candidiasis* in the medical domain. Even though the common approach in lexicography is to enumerate derived words as separate lexical entries, lexicons are often incomplete in practice due to the high productivity of affixes. Thus, grammar-based approaches to detecting the relatedness of derived terms can be useful, as when matching the label *fetishism* with *fetishistic disorder*.

Word inflection. Inflection rules are defined by general language; yet, particular inflected forms can be more or less specific to domains. A well-known example are cooking recipes where sentences tend to begin with verbs either in infinitive or imperative form (e.g. *'Peel the onions'*, which in French may be expressed either as *'Peler les ognons'* or as *'Pelez les ognons'*).

Specific uses of punctuation. In labels of the International Classification of Diseases (ICD), such as *'Hallucinogen use, unspecified with hallucinogen persisting perception disorder (flashbacks)'*, parentheses are used to provide clues for the interpretation of the label. Square brackets, commas, or parentheses are also widely used in ontologies, classifications, and data schemas, such as to provide units of measure for numerical values: *speed (km/h)*. The precise interpretation (e.g. relevance or not with respect to the matching task) of such punctuation and the text they delimit depends on the domain and the particular application at hand.

Domain syntax. The same *'Hallucinogen use. . . '* example from above shows that labels can use non-standard syntax. This is sometimes motivated by the context of use, such as the need to sort the labels alphabetically motivates the use of the adjective *unspecified* in a postpositive form. The phrase *hallucinogen persisting*

perception disorder, on the other hand, includes syntax that is not considered as standard in general language but is common in medical text. While syntax may play a minor role in matching very short labels, for longer classification entries it may be taken into account by the matcher tool, as in the case of SMATCH that performs syntactic parsing.

3 Leveraging Domain Language for Ontology Matching

The hypothesis verified in this paper is that "*matching performance can be improved by relying on knowledge that is specific to domain language*". However, as domain language also incorporates elements of general language, our study also considers this aspect. Accordingly, we classify linguistic background knowledge with respect to being general or domain-specific, as well as with respect to being lexical or grammatical. This delineates the following four categories of knowledge: (1) general lexicon; (2) general grammar; (3) domain lexicon; and (4) domain grammar. Furthermore, we consider three different forms of grammatical knowledge with respect to the linguistic elements to which they apply: (a) phrase-level (syntax, dealing with the way words are organised within labels); (b) word-level (morphology, i.e. grammar that deals with the structure of words); and (c) character-level (e.g. orthography and use of punctuation). Due to the shortness of ontology and classification labels, we deem it sufficient to consider only these three levels of granularity of grammar.

General Lexicon A domain-independent resource that is commonly used is Princeton WordNet [12] which is a lexical database in which nouns, verbs, adjectives and adverbs are grouped into sets of synonyms, each expressing a different concept. All sets are semantically related between them with an *is_a* relationship, forming a taxonomy, in which the more general elements are at the top and the more specific are at the bottom levels.

Domain Lexicon There are multiple domain-specific resources such as lexicons or domain terminologies that contain the technical terms of an specific domain. In the literature, we can find different approaches to integrating these resources within WordNet [1, 17]. Their main goal is to append specialised knowledge to general knowledge currently represented in WordNet (e.g. *coronavirus* as a specialised type of *infection*). However there are cases in which the current representation of a word in WordNet differs from its meaning in the domain-specific resource (e.g. *evacuation* in WordNet and in the UKCP). In these cases, the integration is more complex and needs to be done in a supervised way [17].

The main advantage of using domain lexical knowledge is that matchers have an enriched BK and are able to find mappings of labels that include some of the added new terms. Moreover, when matching ontologies from multiple or partially different domains (such as reference health knowledge involving subdomains of healthcare), domain information can be leveraged for word sense disambiguation within the matching process, resulting in improved precision [5].

General Grammar Most matchers consider the grammar within labels for the matching process. In this case, they carry out some of the following tasks with independence from the domain of the resources to be matched [10].

- *Phrase Level Grammar*.
  - *Tokenisation.* Labels are segmented into tokens (e.g. "*medium-scale evacuation*" becomes <*medium, scale, evacuation*>).
  - *Acronym extraction.* Characters of tokens are used to extract/discover acronyms (e.g. "*Non Governmental Organisation*" becomes "*NGO*").
  - *String similarity.* Compare string labels considering different measures and return a value according to their similarity degree (e.g. "*Level of emergency*" and "*Level 1 emergency*" have a high similarity degree).
  - *Stopword elimination.* Tokens that are recognised as articles, prepositions, conjunctions are removed (e.g. "*level of emergency*" becomes "*level emergency*").
- *Word Level Grammar*.
  - *Lemmatisation.* Tokens are reduced to basic forms (e.g. "*disasters*" becomes "*disaster*").
- *Character Level Grammar*.
  - *Normalisation.* This task includes several subtasks such as: case normalisation, diacritics suppression, blank normalisation, digit suppression or punctuation elimination.

Domain Grammar There are cases in which applying the previous domain-independent tasks to domain-specific resources is counter-productive. For example, if we apply digit suppression and stopword elimination to the following labels: "*Level of emergency*", "*Level 1 emergency*", "*Level 2 emergency*", "*Level 3 emergency*"; the matcher might output that all labels represent the same knowledge. Another example appears when the case normalisation task is just limited to transform all characters within the label into lower case letters. In this case, if the label contains Roman numerals they might pass unnoticed after the case normalisation. For these reasons, it is necessary to consider domain-specific grammar and address it conscientiously. Below there are described the approaches that we have implemented in our research:

- *Phrase Level Grammar*. Finding clues or postscripts that recurrently appear within the labels in a domain is not unusual. In this case, it is necessary to analyse if they add enough knowledge to keep them in the label or it is worth suppressing them (e.g. "Mild cognitive impairment, so-stated").
- *Word Level Grammar*. Implementing derivational morphology rules to transform a term from one part-of-speech into another is interesting because enriching matchers' BK with these words allows those matchers that do not mainly base the matching process on string similarity measures to discover new mappings. Domain words produced by derivational morphology are added to matchers' BK as related forms (e.g. "*pathological*" is added as a related form of "*pathology*).

– *Character Level Grammar*. Depending on the domain, particularly in application domain knowledge resources, orthography follows different conventions. This makes necessary to address it optimally in each case. For example, there might be cases in which the content within parentheses or square brackets is meta-information that is not relevant for the meaning of the label (e.g. "*Post(-) traumatic stress disorder*"), being recommendable its suppression, whereas in other cases this content might be essential (e.g. "*Stable lodine (Potassium lodate tables)*").

The rules of the different domain grammar levels can be extracted both in a supervised or unsupervised way. The latter requires a huge number of documents to apply statistical methods, whereas the former does not need such quantity of documents, but involves more effort. In general, the rules at the *word level* can be transferred to any ontology within a domain (e.g. health), while the rules at the *phrase and character levels* usually are more dependent on the application domain (e.g. Hospitals of North London).

4 Case Study on the Health Domain

The main motivation lies in the need of solving semantic interoperability problems within the health domain. For example, when clinicians have to exchange health records that contain descriptions from multiple official classifications of diseases. To do so, we have developed several extensions to enrich the matcher's BK with health lexical and grammatical knowledge.

Due to descriptions of disorders containing not only technical, but also general terms, WordNet has been used as a domain-independent BK into which the extensions are plugged. The extensions have been developed following the Lexical Markup Framework (LMF) standard [13], and integrated into WordNet using Diversicon [4], which is a framework that allows extending WordNet with any domain-specific knowledge represented in LMF, validating and generating an enriched WordNet.

General Lexicon Princeton WordNet has been used as domain-independent resource. The main reason is that it represents general knowledge and there are multiple approaches that we could apply to enrich WordNet with domain-specific knowledge resources.

Health Domain Lexicon We have developed an extension for WordNet that includes health lexical knowledge extracted from the following resources:

– *MeSH* is the National Library of Medicine's controlled vocabulary thesaurus [16]. It consists of sets of terms, naming descriptions, in a hierarchical structure that permits searching at various levels of specificity. The hierarchy is sorted considering several semantic relations such as *is_a* or *part_of*. This hierarchy is similar to the way in which WordNet is organised, which makes easier its integration. The developed extension for WordNet contains all

descriptions included in the "*Diseases*" and "*Psychiatry and Psychology*" MeSH categories. In this case, we only consider the *is_a* semantic relation, because we have detected several problems using *part_of* when matching diseases (e.g. a "*heel disease*" is a "*foot disease*", but an "*eye disease*" is not a "*face disease*"). Addressing these problems is something that we are considering as a future work.

– The *SPECIALIST* lexicon is an English lexicon which contains both commonly occurring English words and biomedical vocabulary [7]. It is composed of lexical records, being each of them formed by a base form and a set of spelling variants or morphological derivations. For example, the lexical entry with base "*nephroprotective*" (adj) has as spelling variant: "*nephro-protective*", and as morphological derivation "*nephroprotectivity*" (noun). This resource has been used for enriching matchers' BK lexically, through developing an extension for WordNet that contains all lexical entries included in SPECIALIST.

General Grammar It has been addressed applying the grammatical techniques included in the matchers by default and including general derivational morphology.

Phrase level Grammar. The tasks applied have been: tokenisation, string similarity and stop word elimination.

Word level Grammar. In this case we applied lemmatisation and the integration of general derivational morphology rules included in SPECIALIST. Table 1 shows examples of these rules.

Table 1. General derivational morphology rules.

Derivational rule	**Example**
iency$(noun) → ient$(adj)	immuno-deficiency(noun) → immuno-deficient(adj)
sation$(noun) → zed$(adj)	anesthetisation(noun) → anesthetized(adj)
ical$(adj) → y$(noun)	uroradiological(adj) → uroradiology(noun)
ism$(noun) → istic$(adj)	fetichism(noun) → fetichistic(adj)

Character level Grammar. The tasks applied have been case normalisation, blank normalisation and diacritics suppression.

Health Domain Grammar It has been addressed using health derivational morphology extracted from SPECIALIST, and considerations identified at *phrase* and *character grammar levels*. The former was used to enrich matchers' BK, whereas the latter were considered as a preprocessing step prior to the OM process.

Phrase level Grammar. In medical resources there are clues that recurrently appear within descriptions of disorders. Examples are "*, undefined*" and "*, so stated*". This meta-information does not add special value to labels, particularly affecting to those matchers that mainly use string similarity measures. The main reason is that they are penalised by irrelevant characters, which results in a lower

similarity degree. Considering the previous issue we decided to suppress these interpretational clues from descriptions of diseases in a preprocessing step prior to the matching process.

Word level Grammar. Several domain-specific derivational morphology rules have been extracted from the SPECIALIST lexicon and integrated into WordNet. Examples of these rules are shown in table 2.

Table 2. Health derivational morphology rules.

Derivational rule	**Example**
ose$(verb)→ osis$(noun)	sclerose(verb) → sclerosis(noun)
physeal$(adj) → physis$(noun)	adenohypophyseal(adj) → adenohypophysis(noun)
sis$(noun)→ ze$(verb)	dialysis(noun) → dialyze(verb)
a$(noun)→ iasis$(noun)	candida(noun) → candidiasis(noun)

Character level Grammar. We have identified a particular use of parentheses, square brackets and commas in the health domain. Examples of the use of parentheses and square brackets might be the following:

1. Sleep terrors [night terrors]
2. No Diagnosis or Condition on Axis I / No Diagnosis on Axis II [DSM-IV]
3. Premature (early) ejaculation
4. Trichotillomania (hair-pulling disorder)
5. Obstructive sleep apnea (adult) (pediatric)

In case 1, the square brackets are used to specify an equivalent expression of "*sleep terrors*". Similarly, in case 3 parentheses are used to indicate a synonym of "*premature*". Case 2 is different as brackets are used to point out the DSM version in which the description was included. In case 4 the content within parentheses categorises the kind of disorder that "*trichotillomania*" is. Finally, case 5 uses parentheses to indicate the domain to which the disorder is applicable, in that case to *adults* and *children*.

Similarly as in the previous cases, commas are utilised with different purposes in the medical knowledge. Below there are some examples:

1. Tobacco use disorder, Mild
2. Adverse effect of unspecified antidepressants, sequela
3. Circadian rhythm sleep disorder, shift work

In example 1, the comma is used to specify the degree of the disorder, whereas in example 2, it is used to define the kind of adverse effect. Finally, in example 3, the comma is used to specify the cause of the disorder.

This diverse use of parentheses, square brackets and commas, complicates labels, penalising matchers' performance. Thus, we decided to suppress commas and all content within parentheses and square brackets to avoid this penalisation. This simplifies labels and reduces irrelevant content. Nonetheless, in the future, we should investigate less aggressive solutions to reduce matchers penalisation while taking advantage of the content within parentheses.

5 Evaluation

The hypothesis has been evaluated by an experiment in which matchers with different configurations had to match several descriptions of the two most important classifications of diseases for mental health: the Diagnostic and Statistical Manual of Mental Disorders, fifth edition (DSM-5) and the ICD-10. To evaluate the quality of the matchers, we used as gold standard the correspondences between both classifications published in DSM-5, where it is specified to which code in ICD-10 corresponds each description in DSM-5.

The input schemas were a source dataset with 200 entries randomly selected from DSM-5, and a target dataset with 177 descriptions included in ICD-10, which are the correspondences of the entries chosen from DSM-5.

The matchers selected were S-Match [14] and LogMap [15]. The main reasons of choosing these two matchers are their differences to carry out the matching process, and the diverse BK they use. Whereas the former carries out semantic matching, the latter is a highly scalable system that has reasoning and diagnosis capabilities allowing it to detect and repair unsatisfiability on the fly [10]. S-Match uses by default WordNet as BK, so it only includes general knowledge, whereas LogMap only incorporates by default biomedical knowledge provided by resources within of the Unified Medical Language System (UMLS). Regarding grammar, both matchers are limited to address general grammar. While S-Match includes tokenisation, lemmatisation and the translation of punctuation marks into logical connectives, LogMap implements string similarity measures, stop words elimination and word stemming.

The experiments were executed 4 times with each matcher, computing the standard metrics within the information retrieval community: *precision, recall* and *f-measure*. Firstly, with the vanilla version, which was our baseline in each case; secondly, with the lexicon extension; thirdly, with the grammar extension, and finally, with both extensions.

Figure 1 and figure 2 depict the results of the experiments executed in S-Match and LogMap, respectively. We can see how both matchers, S-Match and LogMap, improve their performance in terms of f-measure around 20% and 7% respectively. It is also noticeable, that overall both matchers achieve low results which are caused by the nature of the input labels, which on average are descriptions with more than 5-6 words, so this results in complex label formulas and low string similarity values.

Regarding S-Match, the vanilla version only has a general BK and the matcher is penalised mainly for the way in which it manages commas (each comma is considered as a disjunctive operator). This caused a huge number of false positives, which negatively affected precision, but also discovered, as side effect, a high number of correspondences, resulting in the highest recall. An example is the label "*Mild cognitive impairment, so stated*" which is transformed into the following label formula:

$$mild \ \& \ cognitive\ state \ \& \ impairment \mid state$$

From this label formula S-Match computes the following node formula:

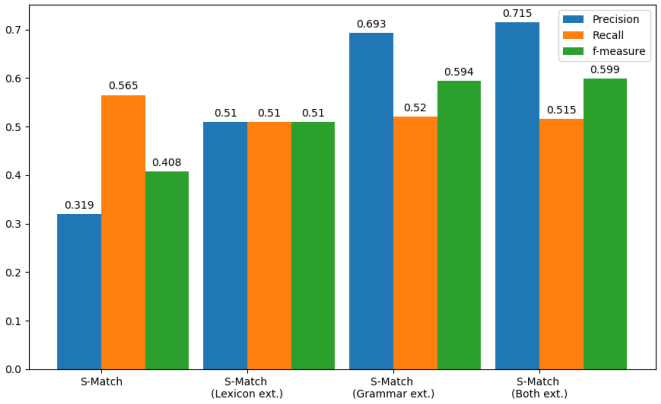


Fig. 1. Results of the experiments executed in S-Match.

$$(mild \mid state) \ \& \ (cognitive\ state \mid state) \ \& \ (impairment \mid state)$$

That means that if "*state*" has a relationship with a lemma within any label of the other ontology, the matcher will output a mapping even if the rest of the label is not related.

The lexicon extension considerably improves the performance (11%) by adding health lexicon knowledge, but this extension also avoids some of the correspondences discovered as side effect, mainly with the inclusion of lexical entries that were considered as single tokens in the vanilla version and now are compound tokens, so the recall slightly decreases.

The grammar extension is the one that drastically reduces the number of false positives mainly with the techniques applied at *phrase* and *character grammar levels* that were employed as a preprocessing step prior to the matching process. In addition, it also discovers new mappings thank to the derivational morphology implemented at *word grammar level*.

The combination of both lexicon and grammar extensions is the configuration that performs better in terms of f-measure, complementing each other and improving the baseline around 20%. However, the false positives of both extensions are also aggregated, being precision slightly penalised.

As for LogMap (see Figure 2), the vanilla version includes biomedical knowledge by default, resulting in a baseline with a performance over 60%.

The lexicon extension added knowledge coming from SPECIALIST, MeSH and WordNet, but it was the latter which produced the major impact as it added domain-independent knowledge contained in the labels. This new knowledge also produced some false positives, but on average this configuration improved the baseline around 6.3%. An example of false positive is:"*Narcolepsy without cataplexy but with hypocretin deficiency*" $\not\equiv$ "*Narcolepsy with cataplexy*", while an example of new true positive is: "*Acute stress disorder*" $\equiv$ "*Acute stress reaction*".

The grammar extension had a similar effect mainly because it also incorporated WordNet. In this case, tasks for *word* and *character grammar levels*

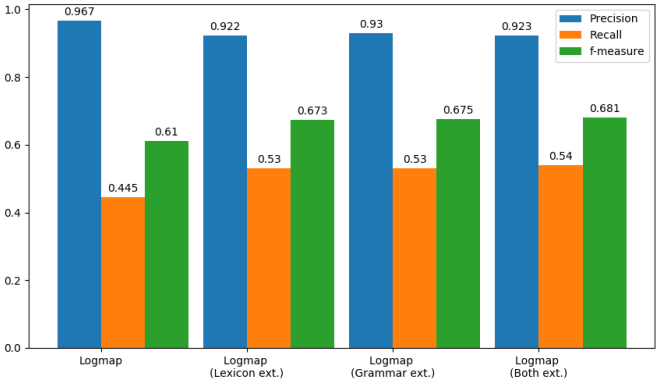


Fig. 2. Results of the experiments executed in LogMap.

had a low impact on LogMap. Nonetheless, *phrase level grammar* preprocessing had a significant impact, and the performance improved 6.5% with respect to the baseline. Examples of new true positives are: "*Trichotillomania (hair-pulling disorder)*" ≡ "*Trichotillomania*", and "*Overweight or obesity*" ≡ "*Obesity, unspecified*".

The combination of both extensions was the configuration that obtained the best performance, achieving the highest number of true positives discovered. In this case, the baseline is improved more than 7%.

6 Concluding Remarks

In this paper, we have presented an approach in which matchers can take advantage of both, domain lexicon and grammar to improve their performance when matching domain-knowledge resources. After evaluating our approach by matching some descriptions of mental health disorders included in DSM-5 and ICD-10 with S-Match and LogMap, we can conclude that our hypothesis is true, as both matchers improve their f-measure compared with the vanilla version.

It is interesting to highlight how the use of domain lexicon and grammar affects differently depending on the matcher. Whereas the domain lexicon extension has the major impact on LogMap, S-Match experiences its major improvement with the grammar extension. The main reason is that LogMap now can discover new mappings thank to domain-independent knowledge, and S-Match has label formulas significantly simplified. This information is useful in order to optimise efforts in the future, and help to decide whether is more valuable investing time focusing on integrating domain lexicon or grammar knowledge into matcher's KB.

As future work we should explore other factors that may affect matchers when matching domain-knowledge, such as the impact of each kind of knowledge represented within knowledge resources according to their levels of specificity. Moreover, it is interesting to delve into methods to aggregate lexicon and grammar results in order to optimise matcher's performance.

Acknowledgements
This research was partially supported by the European Commission with the grant agreement No. 607062 (ESSENCE Marie Curie ITN, http://www.essence-network.com/).

References

1. Amaro, R., Mendes, S.: Towards merging common and technical lexicon wordnets. In: Proceedings of the 3rd Workshop on Cognitive Aspects of the Lexicon. pp. 147–160 (2012)
2. Annane, A., Bellahsene, Z., Azouaou, F., Jonquet, C.: Yam-bio–results for oaei 2017. In: CEUR Workshop Proceedings (2017)
3. Bella, G., Giunchiglia, F., McNeill, F.: Language and domain aware lightweight ontology matching. Journal of Web Semantics **43**, 1–17 (2017)
4. Bella, G., McNeill, F., Leoni, D., Quesada Real, F.J., Giunchiglia, F.: Diversicon: Pluggable lexical domain knowledge. Journal on Data Semantics pp. 1–16 (2019)
5. Bella, G., Zamboni, A., Giunchiglia, F.: Domain-based sense disambiguation in multilingual structured data. In: The Diversity Workshop at the European Conference on Artificial Intelligence (ECAI) (2016)
6. Bella, G., Elliot, L., Das, S., Pavis, S., Turra, E., Robertson, D., Giunchiglia, F.: Cross-border medical research using multi-layered and distributed knowledge (2020)
7. Browne, A.C., McCray, A.T., Srinivasan, S.: The specialist lexicon. National Library of Medicine Technical Reports pp. 18–21 (2000)
8. Da Silva, J., Revoredo, K., Baião, F., Euzenat, J.: A lin: improving interactive ontology matching by interactively revising mapping suggestions. The Knowledge Engineering Review **35** (2020)
9. Dragisic, Z., Ivanova, V., Li, H., Lambrix, P.: Experiences from the anatomy track in the ontology alignment evaluation initiative. Journal of biomedical semantics **8**(1), 56 (2017)
10. Euzenat, J., Shvaiko, P.: Ontology Matching - Second Edition. Springer (2013)
11. Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F., Couto, F.M.: The agreementmakerlight ontology matching system. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 527–541. Springer (2013)
12. Fellbaum, C.: WordNet. Wiley Online Library (1998)
13. Francopoulo, G.: Lmf iso 24613:2008 (Mar 2008)
14. Giunchiglia, F., Shvaiko, P., Yatskevich, M.: S-match: an algorithm and an implementation of semantic matching. In: European semantic web symposium. pp. 61–75. Springer (2004)
15. Jiménez-Ruiz, E., Cuenca Grau, B.: Logmap: Logic-based and scalable ontology matching. In: International Semantic Web Conference. pp. 273–288. Springer (2011)
16. Lipscomb, C.E.: Medical subject headings (mesh). Bulletin of the Medical Library Association **88**(3), 265 (2000)
17. Quesada Real, F.J., McNeill, F., Bella, G., Bundy, A.: Improving dynamic information exchange in emergency response scenarios. In: Proceedings of 14th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2017). pp. 824–833 (2017)
18. Quesada Real, F.J., McNeill, F., Bella, G., Bundy, A.: Identifying semantic domains in emergency scenarios. In: 15th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2018). pp. 1130–1132 (2018)

Semantic Schema Mapping for Interoperable Data-Exchange

Harshvardhan J. Pandit(✉), Damien Graux, Fabrizio Orlandi, Ademar Crotti Junior, Declan O'Sullivan, and Dave Lewis

ADAPT SFI Centre, Trinity College Dublin, Ireland
{pandith,grauxd,orlandif,crottija,declan.osullivan,dave.lewis}@tcd.ie

Abstract. GDPR's Right to Data Portability requires data to be provided in an interoperable, commonly used, and machine-readable format and facilitates its transfer between controllers. However, a major challenge for such data to be used between different services is agreement over common schemas to define the semantics of data. We present our vision of a holistic process for organisations to export and import data in an interoperable manner by using ontology matching and mapping techniques to identify a common model towards schema-negotiation. Our approach enables organisations to exchange data using a common base schema, thereby motivating greater innovation in the ecosystem through data reuse. To demonstrate our vision, we present a proof-of-concept application of ingesting data from Facebook into Twitter.

1 Introduction

Interoperability of data between services can facilitate innovation, collaboration, and competition to enable a richer ecosystem of services. The Right to Data Portability (RtDP) was designed and implemented with this as the motivation in Article 20 of the European General Data Protection Regulation[1] to provide a legal impetus for data to be exported out of silos and shared between services. RtDP requires organisations[2] to provide a copy of personal data they have collected from an individual in a structured, commonly used, and machine-readable format. RtDP also permits data to be transmitted directly to another organisation. In principle, this provides individuals as well as organisations the freedom to obtain and reuse existing data from different services and encourages greater competition and innovation between services by countering data silos and user monopolies.

As of August 2020, however, RtDP is yet to be effectively implemented, and there is a lack of consensus in structure and semantics of data which presents technical difficulties associated with interoperability and data sharing across services [11]. One of the major issues in implementing RtDP concerns the 'semantics' of data i.e. how to indicate the structure, context, and meaning of data

[1] http://data.europa.eu/eli/reg/2016/679/oj

[2] consider 'organisation', Data Controller (GDPR), and 'service' as synonyms in article

in an interoperable form. This issue is further compounded given that GDPR does not mandate use of semantics in provision of data. Therefore, data made under RtDP will either (a) have no schema; or (b) its schema is dictated by the service that exported it. In either case, an individual or organisation that wants to use this data must first understand the structure and contents of the data before building tools to use it – which may be feasible when there are a few services but difficult to scale within an ecosystem.

In this article, we present an overview of practical problems regarding implementation of data portability which skew the balance of power against new services and SMEs (small and medium sized enterprises). We then present our vision for a solution that aims to solve this problem using the notion of semantic interoperability where 'data models' or 'schemas' are a) developed within a community, b) embedded or associated with data to convey meaning, and c) aligned with other schemas to enable importing and exporting data between services – thus achieving the intended goals of RtDP.

The novelty of our approach is within the lack of consensus about semantics required between exporting and importing services through a registry of curated schemas that act as a base for interpretation and permit variations in use-cases and applications. To achieve this vision, we propose the use of ontology matching and alignment techniques as the 'bridge' for data interoperability between two services. Further, we discuss the application and role of ontology matching to produce mappings for exporting (downlift) and importing (uplift) data directly between services.

The rest of this article is structured as follows: Section 2 presents the legal requirements and existing implementations of RtDP, and discusses practical challenges with a focus on the feasibility of meaningful exchange of data and the role of semantics; Section 3 presents our vision of a solution and its application on a hypothetical scenario involving transfer of data from Facebook to Twitter; Section 4 concludes this article with a discussion on the practical considerations for implementing our solution and its potential for helping SMEs innovate in an existing dominant ecosystem.

2 RtDP in the Real-World

2.1 GDPR Requirements, Authoritative Opinions, and Guidelines

Article 20 and Recital 68 of the GDPR[3] stipulate data to be provided under RtDP to be structured, commonly used, machine-readable, and interoperable format. further introduces the requirement of interoperability and motivates creation of interoperable formats that enable data portability. They also provide for such data to be transferred (directly) from one Data Controller to another. The guidelines on RtDP provided by Article 29 Working Party (WP29) further

[3] This articles focuses only on the data formats and interoperability requirements for RtDP. Conditions where the right applies, obligations of an organisation, and its compliance is not relevant to this work.

clarify that the RtDP "does not place obligations on other data controllers to support these formats" [5].

Guidelines by WP29 and various Data Protection Authorities on data formats includes use of XML, JSON, and CSV which are widely adopted and used for interoperability. WP29 states that such data formats should be accompanied "with useful metadata at the best possible level of granularity, while maintaining a high level of abstraction ... in order to accurately describe the meaning of exchanged information" [5]. ICO, which is the Data Protection Authority for UK, explicitly suggests RDF[4] as a standardised data format for interoperability. Thus, although the GDPR motivates data sharing between services, it only suggests semantic interoperability[5] with RDF being a practical solution.

Currently, EU's Next Generation Internet initiative is funding projects through the Data Portability and Services Incubator (DAPSI[6]) which lists directions for possible solutions as common shared formats, vocabularies and ontologies for domains, and methods for (semi-)automatically converting data including semantic mapping. The ISO/IEC 19941:2017[7] standard for cloud interoperability outlines the requirements for semantic interoperability, and the practical use of semantic web standards towards shared understanding. An early paper from 2008 presented reuse of semantic web vocabularies for data interoperability within social networks [1]. This shows that the semantic web domain has been a known direction for a solution towards effective implementation of RtDP and achieving semantic interoperability.

2.2 Real-world Implementations

RtDP has been implemented in a wide range of services given its nature as a legal obligation. Several organisations have developed dedicated tools for RtDP such as Google's 'Takeout', Facebook's 'Download Your Information', and Twitter's 'Your Twitter Data'. An example of data portability directly between services is transferring photos from Facebook to Google Photos[8]. The Data Transfer Project[9] (DTP) is a combined initiative consisting of IT behemoths Apple, Facebook, Google, Microsoft, Twitter - to develop an open-source, service-to-service data portability platform. To this end the project is developing[10] 'Data Models' as a common interoperable schema between services.

While these examples are optimistic, the reality is that RtDP has not seen its full impact, and has not been sufficiently implemented by any service or organi-

[4] https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/individual-rights/right-to-data-portability/

[5] Semantic interoperability was an explicit aim in earlier drafts of WP29 guidelines but was reduced to just 'interoperability' in the final published version [3]

[6] https://dapsi.ngi.eu/

[7] https://www.iso.org/standard/66639.html

[8] https://about.fb.com/news/2019/12/data-portability-photo-transfer-tool/

[9] https://datatransferproject.dev/

[10] https://github.com/google/data-transfer-project/

sation. A survey of data formats used in RtDP [10] shows variation in responses, non-conformance with GDPR requirements, and a lack of semantics. The Data Transfer Project, though it has been running for over 2 years (2018-2020), has not produced any usable results to achieve its aims despite involving the worlds largest IT organisations. An article by De Hert et al. [3] outlines the challenges in implementing RtDP with two potential approaches: (i) minimalist approach - which requires organisations to minimally comply with the GDPR; and (ii) empowering approach - where semantic interoperability provides a stimulus of choice and freedom to the user along with encouraging competition and innovation amongst services. It is the nature of free-market capitalism that established players prefer (i) whilst users and new entrants would prefer (ii) - each for their own benefit. Our vision thus rests on making possible the empowering approach within an ecosystem without additional obligations on organisations that only want to implement the minimal approach for compliance.

2.3 Challenges in implementing Right to Data Portability

Semantic interoperability, in its role as a solution for data portability, depends on the establishment and sharing of schemas along with the data. *schema.org*[11] is a good example of shared and interoperable schema development across services and use-cases based on its continued development and use at web-scale. Another example is Shape Repo[12] which reuses existing vocabularies (such as WikiData[13]) to declare schemas for use in SOLID[14] application development. Similar to these, we base our approach on establishment of common schemas for semantic interoperability through community engagement and maintenance. In this section, we discuss some challenges present within the ecosystem which justify our approach of a community-driven common schema.

(1) When exported data contains no schema: Unless there is an explicit legal requirement that mandates the association of schemas in a specific manner with exported datasets, this situation is likely to continue. So the question arises over who should develop and maintain the schemas? A dominant organisation has interest in maintaining control over its data and reducing its usefulness to other organisations who might be potential competitors. At the same time, these other organisations (and individuals) would be interested in reusing the exported data to enrich or enhance their own features and offerings. Therefore, it is in the natural interest of the community at large to produce schemas to enrich its data-based services to drive innovation and competition. The existing ecosystem based on services offering APIs presents validation of this argument.

(2) When exported data contains a schema: If a service already provides a schema with its exported dataset, it is easier to utilise this schema rather than develop a new one. However, in the longer run, an independent schema is

[11] https://schema.org/
[12] https://shaperepo.com/
[13] https://www.wikidata.org/
[14] https://solidproject.org/

more resilient to control by one provider and can also be managed more efficiently across use-cases. This is evident in the situation where the service changes its schema, thereby requiring every tool and service dependant on its schema to also change their implementations. Therefore, even where a data comes with a schema attached, it is beneficial to develop a common schema and super-impose the data's schema on it.

(3) Stakeholders beyond domains: Thus far, we have only considered situations where services directly compete with each other within the same domain. However, data can also be useful for integration into other services or for added features. An example of this is a service that offers recording 'daily logs' from a user's social media posts regardless of service. In such cases, it may be to the benefit of the service provider to encourage development of features dependant on its data. While the data providing service would want to restrict such services to only work with their data, the service itself would be inclined to support as many services as possible - an avenue for using common schema and tools based on it.

(4) Cost of development and Control: Larger organisations have more resources at their disposal and larger freedom to experiment. Small organisations (SMEs) are often resource-constrained and rely on innovation to compete. Therefore, a common and shared approach for managing intoperable data is of greater benefit to SMEs, which provides an incentive for them to pool their use-cases and resources together to collaborate and share the burden of competition.

3 Proposed solution

Our vision for implementing RtDP addresses the challenges discussed in Section 2.3 by proposing use of common schemas for 'semantic interoperability' in data exchange between services. This includes an interoperable data portability arrangement that benefits all stakeholders by permitting data exporters to continue using their own semantics and data importers understanding the embedded semantics in data. The common schema is used to abstract service-specific design patterns and to serve as a source for common data within a domain. The shared-community aspect of the approach enables sharing of tasks and reducing the effort required in reuse of data and establishing common schemas.

The role of semantic web in this process concerns acting as an interoperable semantic representation using the RDF, RDFS, and OWL standards. We propose utilising ontology matching and alignment to identify the correct schemas for data exported from service A to be transformed and imported into service B. We also propose utilising ontology matching to permit reuse of data based on common schemas without explicit agreement between an exporter and importer. Similarly, we also propose using uplift/downlift mappings between schemas as a technique to potentially perform this step without requiring transformation of data into RDF.

Ontology matching is "the process of generating an ontology alignment between a source and a target ontology" [4]. In the last 15 years, a number of sur-

veys has been published in the area. They review the various techniques proposed for two main categories of approaches, focusing either on *simple correspondences* between concepts/resources [7][6] (*1:1* concept matching) or *complex matching* [9] (for *m:n* or more complex relations). Since ontology matching is one of the oldest and most relevant research areas in the Semantic Web community[15], it has produced a wide variety of techniques and tools ready to be used[16]. Popular implementations, such as the Alignment API[17] [2] or the NeOn Toolkit[18], assist practitioners in attempting to automatically align different schemas.

To explain and discuss the application of semantic web, ontology matching, and mappings in our approach in detail, consider the hypothetical use-case of an individual wishing to obtain posts exported from Facebook and import them to Twitter. This use-case can also be generalised for services both within and outside the social media domain looking to import and reuse some or all of the Facebook data - which furthers the usefulness of our approach.

3.1 Data Ingestion & Conversion

Currently, both Facebook and Twitter[19] export their data under RtDP as JSON[20] — a non-semantic format.

The first step in ingesting Facebook's JSON data is thus to understand its structure and its schema. Where services undertake this effort individually, each service has to duplicate the effort of understanding the structure and keeping its tool updated. By sharing this task, the community can maintain a documentation of the data's schema and structure. If and when Facebook changes the data structure or format, the community can update its documentation without duplication of effort. While it is Facebook's prerogative to structure its data and change it as it feels fit - an argument can be made that frequent and unreasonable changes are detrimental to the spirit of RtDP.

To minimise impact of such changes, a schema corresponding to Facebook's data is created in the common registry, and any tools ingesting Facebook's data utilise the schema instead. Minimal effort is required to 'transform' the data from its underlying structure to one corresponding with the established schema - such as through a python script to convert to CSV or through RDF mapping to convert to JSNO-LD - based on what the desired output format is.

[15] The "OM" workshop has been continuously running at ISWC since 2006.
[16] OAEI, the Ontology Alignment Evaluation Initiative, has been running yearly since 2004, evaluating the latest ontology matching technologies: http://oaei.ontologymatching.org/
[17] http://alignapi.gforge.inria.fr/
[18] http://neon-toolkit.org/
[19] Information about Twitter's data may be out-of-date as its export tool has been non-operational as of August-15-2020.
[20] Facebook exports data as a JSON dump. Twitter exports data as a JavaScript file with JSON objects. Neither supply information about the schema or structure of their data.

3.2 Schema Description

The creation of a Facebook schema is based on first creating a common schema representing 'a social media post'. The concepts in the Facebook schema are thus specialised variations of the common schema, representing Facebook as a specific type of social media. This abstraction permits a data importer to target data specifically from Facebook (through the Facebook schema) or any social media (through the common social media schema). The abstraction also works to encourage designing common tools to work on the data rather than specialised ones targeting individual services. Figure 1 depicts an example of a common schema targeting social media posts.

The creation of a common schema where none exists is difficult if a community agreement is necessary over its concepts and structure. Therefore, we suggest seeding the first common schema with concepts from dominant data providers in the domain and normalising it towards other existing providers. In the current use-case, this would mean first creating a schema from Facebook's data, then creating a common schema based on Facebook's schema, and updating Facebook's schema to use the common one as its base. By this we mean sub-classing concepts in specialised schemas from common ones. Later, when creating Twitter's schema, the existing common schema for social media can be used to guide the schema creation process.

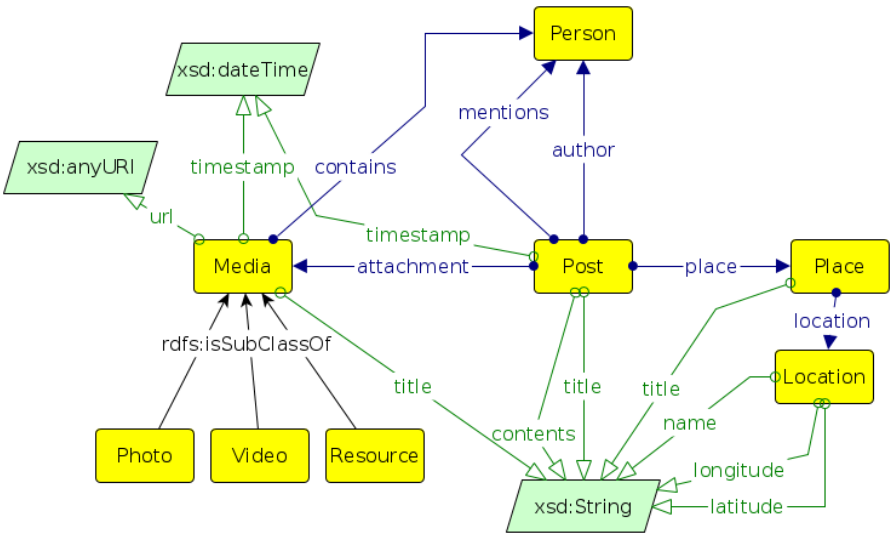


Fig. 1. Example of a common schema for social media post.

3.3 Schema Alignment

In the common and Facebook schemas, the generic terms 'post', 'media', 'timestamp' are suitable for use in both since Facebook does not have any specialised variation of these. However, concepts such as 'like' or 'reaction' may present problems in abstraction and generalisation as they may not be present in other service in the same context. For example, Twitter defines[21] the '♥' symbol to mean a 'like' whereas Facebook defines[22] its 'reactions' as an enumeration consisting of 'like, love, wow, haha, sorry, angry'. Aligning the two is difficult due to semantic differences in the two terms. One interpretation is that only a Facebook 'love' is equivalent to Twitter 'like', whereas another possible interpretation is that any Facebook reaction should be equivalent to Twitter 'like'.

We propose the use of ontology matching and alignment techniques to assist in the schema alignment and discovery process as well as to resolve equivalence between concepts. This can be an automated process, but we also emphasise its value in encouraging discussion amongst schema creators and maintainers through a human-in-the-loop process. The role of common schemas in this is to provide a measure of commonality in the identification and structuring of source and target schemas, as well as to ease the process of finding related and equivalent data patterns. For example, in the case of a Facebook post and Twitter 'tweet', the relationship is easy to establish based on their common super-classes.

Facebook	**Common Schema**	**Twitter**	**Type of alignment**
Post	Post	Tweet	Simple
Contents	Contents	Contents	Simple
Timestamp	Timestamp	Timestamp	Simple
User	Person	Profile	Complex
Friend	Knows	Follows	Complex
Attachment	Media	Media	Simple

Ontology alignment techniques may also provide a way to integrate data where no possible contextual similarity is apparent. For example, Facebook's 'friend' concept and Twitter's 'follows' concept are different in their behaviour and discourse - yet they share similarity in their pattern of association with an individual. It is up to the importer then to determine whether they want to support and utilise such alignments or to discard them in favour of more semantically-refined ones.

Once the matching concepts have been found, the process of transferring data to the target service can take place. An explicit way to do this is to first transform the source data to RDF using its corresponding schema (in this case the Facebook schema), then creating an alignment table using the ontology matching process, and then to generate the dataset using the target schema (in this

[21] https://help.twitter.com/en/glossary

[22] https://developers.facebook.com/docs/graph-api/reference/v8.0/object/reactions

case the Twitter schema). To reduce the number of transformations required in this process, mappings can be potentially used to directly enable the importing service to ingest the source data without the intermediary transformations.

Uplift mapping is the process of converting a data into RDF, while downlift is its inverse. Considering that Facebook exports a JSON data dump, and that Twitter similarly will import[23] a JSON data dump - the process of transformations will involve: (i) uplift Facebook's JSON data into RDF using Facebook schema; (ii) transform RDF data from source schema into target schema using the ontology mapping process; (iii) downlift data into JSON for Twitter. Since the role of step (ii) is merely to find an alignment between the schemas of Facebook and Twitter, the actual transformation of data can take place directly from Facebook's JSON to Twitter's JSON format.

3.4 Using mappings to automate the process

An interesting research question thus arises out of this arrangement - "can we utilise the schema alignments and the mappings to create a tool that will convert the source data to target data?". We believe that it is reasonable to hypothesise that such a tool can indeed be created based on the fact that the structure (i.e. specific arrangement of data structures) of source and target data can itself be considered schemas, and therefore can be utilised to convert one to another. The question around implementing this is then concerned about the efficiency rather than sufficiency. A related area facing similar research challenges is the utilisation of GraphQL to retrieve data from a triple-store in the shape requested by rewriting the query in SPARQL [8].

The use-case we discussed concerned moving data from one social media service to another (Facebook to Twitter). However, RtDP makes it possible to reuse data across a larger plethora of services across domains. For example, Facebook's data contains information about locations the user has tagged their post with (or checked-in). This information could be relevant in any other service providing features that utilise location data - such as a visualisation service that shows all the locations an user has been to on a map. Such a service may want to broaden its data import feature to encourage users to submit *any* location data regardless of its source. Potential sources of such data include: explicit location data shared by user, location tagged in photos, location tagged in social media posts, location inferred from place names and descriptions, location associated with review of a restaurant, or location associated with monetary transactions of a card. Instead of developing separate tools for each of these sources, the service can instead target the underlying common location schema and utilise our approach to ingest the data from a variety of source without additional effort.

In order to identify the potential sources of data, the service can declare the schema for the data it intends to import. For example, this can be a location

[23] Twitter does not provide a data import service. So we reasonably assume its import tool will accept the same data format and structure as its export tool

concept with a label and co-ordinates. A label-based search for related schemas will only retrieve schemas that contain the concept location or its synonym such as 'place'. However, ontology matching techniques can provide richer results by identifying similarly 'shaped schemas' that contain labels and co-ordinates. Further fine tuning is possible by focusing on co-ordinates and its variations while excluding labels. This thus provides an opportunity for utilising ontology matching techniques to identify relevant design patterns for schema discovery.

4 Conclusion

In this paper, we proposed an approach leveraging ontology matching and alignment techniques to achieve data interoperability between online services dealing with personal data. Having GDPR's Right to Data Portability (RtDP) in mind, we described a typical use-case where users of a social networking service (e.g. Facebook & Twitter) are willing to — and should be allowed to — export their own personal data in a machine-readable format and reuse it on a different service. We described how Semantic Web technologies and ontology matching could assist in the alignment with a common schema that is used as a 'bridge' between heterogeneous data schemas. The role of common schemas is to provide a measure of commonality in the structuring of source and target schemas. Finally, we showed how data mappings could be used, and shared via a community-driven repository, to automate the conversion processes. Actually, this last point opens the doors of efficient Data Portability to SMEs which have to allow this feature given the RtDP; in particular, SMEs will be able to minimise the cost of making user data more easily ported to another provider.

We envisage several advantages with the adoption of the proposed approach, both for end-users and companies. *First*, schemas and mappings are open and maintained by the community, lowering the costs for both parties in managing the data transformations. *Second*, maintenance costs are lowered and distributed to the community, removing possible bottlenecks or single points of failure, typical of ad-hoc data transformation pipelines. *Third*, a descriptive and machine-readable schema would not be required from the data exporters anymore, keeping the complexity low at the data sources. *Fourth*, reliability of data transformations would increase. For instance, when one data source changes, mappings updates are faster to perform compared to changes to many ad-hoc pipelines. *Fifth*, the automation potential would increase dramatically with improved, more accurate, ontology matching techniques.

As part of our future work, we plan to implement and test our solution in different use-cases and with different services. This would create a baseline that can be offered to the community and, ideally, adopted and expanded by the community itself. From a more scientific perspective, we will investigate the increased automation possibilities offered by complex ontology matching techniques. Other avenues of potential work include exploration of our approach for interoperability between services and APIs based on semantics, evaluating

the efficiency and feasibility at large scales, and discussing the application of our approach within the broader areas of legal compliance and data protection.

Acknowledgments: This research was conducted with the financial support of the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreements No. 801522 and No. 713567 at the ADAPT SFI Research Centre at Trinity College Dublin. The ADAPT SFI Centre for Digital Media Technology is funded by Science Foundation Ireland through the SFI Research Centres Programme and is co-funded under the European Regional Development Fund (ERDF) through Grant #13/RC/2106.

References

1. Boja, U.: Social Network and Data Portability using Semantic Web Technologies. In: Social Aspects of the Web (SAW 2008), Advances in Accessing Deep Web (ADW 2008), E-Learning for Business Needs. p. 15 (May 2008)
2. David, J., Euzenat, J., Scharffe, F., dos Santos, C.T.: The alignment api 4.0. Semantic Web **2**, 3–10 (2011)
3. De Hert, P., Papakonstantinou, V., Malgieri, G., Beslay, L., Sanchez, I.: The right to data portability in the GDPR: Towards user-centric interoperability of digital services. Computer Law & Security Review **34**(2), 193–203 (Apr 2018). https://doi.org/10/gdtmx7
4. Euzenat, J., Shvaiko, P., et al.: Ontology matching, vol. 18. Springer (2007)
5. Guidelines on the right to data portability 16/EN WP 242 rev.01. Article 29 Data Protection Working Party (Dec 2016)
6. Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. Expert Syst. Appl. **42**(2), 949–971 (2015), https://www.sciencedirect.com/science/article/pii/S0957417414005144
7. Shvaiko, P., Euzenat, J.: A survey of schema-based matching approaches. In: J. Data Semantics IV (2005)
8. Taelman, R., Vander Sande, M., Verborgh, R.: GraphQL-LD: Linked Data querying with GraphQL. In: Proceedings of the 17th International Semantic Web Conference: Posters and Demos (Oct 2018), https://comunica.github.io/Article-ISWC2018-Demo-GraphQlLD/
9. Thiéblin, E., Haemmerlé, O., Hernandez, N., Trojahn, C.: Survey on complex ontology matching. Semantic Web pp. 1–39 (Oct 2019). https://doi.org/10/gg6rd4
10. Wong, J., Henderson, T.: How Portable is Portable?: Exercising the GDPR's Right to Data Portability. In: Proceedings of the 2018 ACM International Joint Conference and 2018 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers. pp. 911–920. UbiComp '18, ACM, New York, NY, USA (2018). https://doi.org/10/gfsqrk
11. Zichichi, M., Rodrıguez-Doncel, V., Ferretti, S.: The use of Decentralized and Semantic Web Technologies for Personal Data Protection and Interoperability. In: GDPR Compliance - Theories, Techniques, Tools Workshop of Jurix 2019. p. 10 (2019)

A Gold Standard Dataset for Large Knowledge Graphs Matching

Omaima Fallatah[1,2][0000−0002−5466−9119], Ziqi Zhang[1][0000−0002−8587−8618], and Frank Hopfgartner[1][0000−0003−0380−6088]

[1] Information School, The University of Sheffield, Sheffield, UK
{oafallatah1,ziqi.zhang,f.hopfgartner}@sheffield.ac.uk
[2] Department of Information Systems, Umm Al Qura University, Saudi Arabia
oafallatah@uqu.edu.sa

Abstract. In the last decade, a remarkable number of Knowledge Graphs (KGs) were developed, such as DBpedia, NELL and Google knowledge graph. These KGs are the core of many web-based applications such as query answering and semantic web navigation. The majority of these KGs are semi-automatically constructed, which has resulted in a significant degree of heterogeneity. KGs are highly complementary; thus, mapping them can benefit intelligent applications that require integrating different KGs such as recommendation systems and search engines. Although the problem of ontology matching has been investigated and a significant number of systems have been developed, the challenges of mapping large-scale KGs remain significant. In 2018, OAEI has introduced a specific track for KG matching systems. Nonetheless, a major limitation of the current benchmark is their lack of representation of real-world KGs. In this work we introduce a gold standard dataset for matching the schema of large, automatically constructed, less-well structured KGs based on DBpedia and NELL. We evaluate OAEI's various participating systems on this dataset, and show that matching large-scale and domain independent KGs is a more challenging task. We believe that the dataset which we make public in this work makes the largest domain-independent gold standard dataset for matching KG classes.

Keywords: Knowledge Graphs · Schema Matching · Evaluation Dataset.

1 Introduction

In the last decade, different KGs have been created as a result of years of information extraction practices and crowdsourcing. DBpedia [3], YAGO [20], and NELL [4] are examples of large domain-independent KGs. Such KGs cover multiple domains of knowledge such as medical, music, and publications. KGs play a significant role in many applications such as reasoning, search engines and e-commerce, while also being part of the linked open data domain [17].

Moreover, due to their automatically-constructed and independently-designed nature, such KGs contain overlapping and complementary facts. For instance, `Bone` and `Artery` are classified under `BodyPart` in NELL while being classified as `AnatomicalStructure` in DBpedia.

This problem of semantic heterogeneity has been thoroughly studied in the Semantic Web community, with many ontology matching systems being developed and surveyed [2]. Matching systems are annually evaluated through the Ontology Alignment Evaluation Initiative (OAEI)[3]. While a new track for KG matching has been introduced to OAEI's annual campaign since 2018, the challenges of aligning large-scale KGs remain significant [12]. Currently, existing gold standards are not well representative of real-world KGs. Such KGs are known for sharing complementary facts about real-world entities such as people and places, while current datasets are predominantly domain-dependent [1]. Further, the size of the existing gold standard does not accurately represent the complexity of matching large-scale KGs that imply a significantly larger search space due to orders of magnitude larger number of classes.

This work proposes a gold standard dataset for matching the classes of large, automatically constructed, inadequately structured, and domain-independent KGs. The introduced benchmark is based on DBpedia and NELL. Although both KGs are widely used in semantic web researches and can be considered highly influential, they are yet to be consolidated, even though the majority of LOD cross-domain datasets, including KGs, are interlinked to DBpedia[4] which serves as a central link to many LOD datasets. According to [19], NELL is considered as the most complementary KG to other larger KGs such as DBpedia with an average of 10% gain of instances, while merging other large KGs can only lead to a 5% gain. Therefore, we believe they are the best candidates for a gold standard dataset for aligning large cross-domain KGs. We conduct an experiment to evaluate the performance of OAEI's different participating systems on this dataset, and show that mapping the classes of open KG is a much more challenging task than the existing OAEI KG matching benchmark.

The rest of this paper is structured as follows. We start by reviewing the problem, and the current gold standard datasets for matching KGs in Section 2. Then, we describe the process of building the proposed dataset in Section 3. In Section 4, we present the results of evaluating current matching systems on the proposed gold standard . We close with a discussion and a conclusion in Sections 5 and 6 respectively.

2 Related Work

Ontology matching has been a well-studied problem which centers on discovering corresponding entities across two distinct ontologies [8]. In the last decade, many matching systems were developed and evaluated annually at the OAEI event.

[3] `http://oaei.ontologymatching.org/`
[4] `https://lod-cloud.net`

The initiative provides over ten benchmark datasets in different tracks for various matching systems to be evaluated. Examples of main tracks are Anatomy, Conference, Complex Matching, Large Biomedical, and Interactive Matching.

KGs are often compared to ontologies since both are used for data representation purposes. Different from former ontology, open KGs are large-scale, multi-domain and less well-formatted compared to ontologies [22]. Similar to ontologies, KGs entities also suffer from semantic heterogeneity where the same real-world entities are described using different terminologies.

While there have been many well established matching systems for OAEI's different matching tracks, the need for KG matchers remains an open area of research [12]. Research in this domain has only been established since 2018, when OAEI introduced a new track dedicated to KG matching[5]. Since then, ontology matching tools have been evaluated on the provided benchmark, and multiple KG matchers have participated in the latest version in 2019 [1]. Although matching KGs has been a growing area of research recently, there is still a lack of gold standard datasets that represent diverse KGs.

The benchmark dataset currently used to evaluate systems in OAEI's KG track is constructed from DBkWik [11], which is a KG created from wikis shared on a wiki hosting platform. The individual KGs from the DBkWik project were used to create the ground truth datasets for this track. The track consists of five test cases where each test case is aimed at matching both the schema, including classes and properties, and the instance level of two KGs. The schema level correspondences were built by ontology experts while the instance level correspondences were automatically extracted [1]. To the best of our knowledge this gold standard is the only benchmark available to evaluate KG matching systems. However, the number of mapped classes is considerably small, i.e., less than 50 [12]. Therefore, this dataset does not represent the complexity of matching real-world KGs where hundreds of classes can be matched.

In terms of large domain-independent KGs, there are many published according to the Semantic Web standards. Some of them are based on Wikipedia, such as YAGO and DBpedia. Originally, DBpedia is a knowledge base constructed from structured data embedded on Wikipedia [3]. DBpedia also involves crowdsourcing communities to maintain the quality of the mapping between Wikipedia's articles and the structured knowledge in their KG. In contrast, NELL is a fully automated learned KG under the Never-Ending Language Learner project, which uses machine learning to read and extract knowledge from free text on the web. It started with a seed KG that continuously evolves by learning patterns from text to extract facts that are used to constantly grow and update the seed KG [4]. Since its launch in 2010, NELL has grown to a KG containing 50 million facts[6]. While the schema of the majority of Wikipedia based KGs cover multiple types of properties, NELL graph schema is very basic. It does not contain as many relations between instances [19]. Another KG

[5] http://oaei.ontologymatching.org/2019/knowledgegraph/index.html
[6] http://rtw.ml.cmu.edu/rtw/

of a taxonomy structure is WebIsALOD [10]. However, the latter only covers hypernymy relations and does not distinguish classes from instances.

3 Approach

3.1 Overview

As mentioned earlier in Section 1, we use NELL and DBpedia as both KGs share a significant amount of complementary facts. In this work, we deploy DBpedia 2016-10 version [7] using SPARQL query endpoint to return schema information. As for NELL, since a query end point is not available we obtained schema information by parsing a NELL dump file[8] which contains every fact learned by the project so far. As a result, DBpedia has over 750 classes while NELL has around 290 classes. Let $\boldsymbol{P}$ be the set of pair-wise classes across the two KGs, then the number of all possible pairs is 218,660. Since our goal is to use human annotators to identify all mappable pairs of classes, a greedy approach will lead to a dataset that is expensive to annotate and likely to be overwhelmed with negative pairs. Instead, we first apply a **Blocking Strategy** to manually generate a set of candidate pairs $\boldsymbol{C}$ which is a subset of $\boldsymbol{P}$ with significantly reduced number of negative class pairs. Next, we perform a **Candidate Filtering Strategy** by applying two similarity measures to each pair in $\boldsymbol{C}$ to further reduce the search space for human annotators. Another screening was done after the filtering stage to ensure that none of the discarded classes had a potential match in the corresponding graph. Finally, for **Dataset Annotation**, we asked human annotators to determine alignment of the resulting class pairs to construct the gold standard dataset.

3.2 Generating Candidate Pairs

Given the two KGs, we set one as source and one as target. Details about our source and target choices will be explained later. Moreover, given $\boldsymbol{P}$, the set of all possible class pairs from the two KGs, we apply a **Blocking Strategy** which requires manually screening the two KG class structures. The result of this process is a set $\boldsymbol{C}$ which should eliminate as many true negatives as possible while maintaining as many as (if not all) true positives. To illustrate the complexity of the task, the classes named `School` in both KGs refer to different types of schools. For instance, it is categorized as a subclass of `EducationalInstitutions` in DBpedia while being a super class of `HighSchool` and `University` classes in NELL. Given this structural inconsistency issue, a preliminary study aimed at aligning the higher level of concepts across the two KGs was necessary.

[7] `https://wiki.dbpedia.org/develop/datasets/dbpedia-version-2016-10`,visited on 14-2-2020

[8] `http://rtw.ml.cmu.edu/rtw/resources`, iteration number 1115, visited on 22-2-2020

We manually created two subsets $\boldsymbol{A}$ and $\boldsymbol{B}$, where the first is a set of NELL classes that have a possible corresponding class in DBpedia, and the second is a set of DBpedia classes that have a possible corresponding class in NELL. These two sets were created in two phases. First, we started by comparing the common root classes across the two KGs, e.g., `Person` or `Place`. Then, all of their non-root (descendant) classes were added to $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively. For instance, all the descendant classes of $Person_{nell}$ and $Person_{dbp}$ were added to each of $\boldsymbol{A}$ and $\boldsymbol{B}$ respectively. Second, we examined other possible classes in which their root classes do not share an overlap of words, i.e., they were not selected in the first step. A valid example are the two classes $AcdemicSubject_{dbp}$ and its possible equivalent class $AcademicField_{nell}$. While the former is a subclass of `TopicalConcept`, the second is a subclass of `everypromotedthing`. The latter is the root class of the KG taxonomic tree, i.e., the equivalent of `OWL:Thing` in DBpedia. Therefore, our second screening phase was aimed at all descendant classes in both KGs whose name values share overlapping words while their super classes do not share overlapping words.

As a result of this blocking strategy, a total of 18,492 candidate pairs were generated in $\boldsymbol{C}$ as the product of $\boldsymbol{A}$ and $\boldsymbol{B}$. We believe this blocking strategy will not incorrectly discard any true positives because we have examined all discarded classes to identify any possible match in the opposite KG. As shown in Table 1, the number of distinct classes from DBpedia and NELL is 138 and 134 respectively. Nonetheless, this number of candidate pairs remains expensive for an annotation task. Therefore, we proceed by the Candidate Filtering Strategy to further reduce the numbers of pairs that need to be annotated by human annotators while maintaining pair completeness.

Table 1. Number of classes and instances in the created dataset

Dataset	#Classes	#Instances	Avg #instance per class
DBpedia	138	631,461	4,576
NELL	134	1,184,377	8,905

3.3 Candidate Filtering

In this section, we introduce the similarity measures applied to the candidate pairs resulting from the prior phase. We apply a string-based and an instance-based similarity measure combined with a low threshold to maximise the chance to retain all true positives. We apply a **String-based Similarity** measure to class names only since NELL does not offer other metadata descriptions of classes. However, using only a string-based matcher can not guarantee a high recall as both KGs use different names to describe the same classes. We then apply an **Instance-based Similarity** measure to capture any possible true positive pairs where string similarity could have failed to recognize them. To the best of our knowledge, a matching approach that can handle a substantial number of instances, such as in the case of KGs , is yet to be established. Therefore,

Section 3.3 discusses the implementation of our preliminary instance-based approach. We believe that combining both measures can ensure high (if not full) recall of true positive pairs. It is also worth mentioning that due to the structural irregularity in both KGs, structural-based similarity measures were excluded.

String-based Similarity Measure We apply the *Levenshtein* [16] edit distance approach. This method has shown improvements over alternative string-based measures, particularly for matching classes [5]. Here the similarity between class names in each candidate pair is measured. This value is then normalized by dividing the value by the length of the longer string, i.e., class name, to produces a value between [0.0, 1.0]. For this task we only retain a pair if the similarity score of the two class names exceeds 0.4. State-of-the-art matching systems that utilize an edit distance approach often apply a higher threshold, which can be up to 0.8, to eliminate the number of false positive alignments [2]. Nonetheless, in order to capture as many true positive pairs as possible, we use a threshold that is twice lower than the state-of-the-art methods.

Instance-based Similarity Measure This method casts the matching process based on the principle of free-text index and search, which scales to very large datasets. On a typical index/search scenario a collection of resources (i.e. web documents) is indexed in a vector space where documents are represented with weighted vectors of their text content. Weighting approaches, such as TF/IDF, are used to weight term occurrences in the documents. A query given to a search engine will also be converted into a vector representation and then matched against all the vectors stored in the index. The matching is done by similarity measures such as the cosine function where a ranked list of top K documents related to the query is retrieved. Similarly, we propose to treat both KGs as a collection of documents where each document corresponds to a class in a KG and each term corresponds to the name of an instance. To map similar classes, a query is built by sampling instance names from a source KG's class, and matching against the index of the target KG. The equivalent class is determined based on the search result, which is a ranked list of classes whose instance names overlap with those in the query. We exploit *Apache Solr*[9], a state-of-the-art free text index and search engine. The pseudocode for the entire similarity measure is illustrated in Algorithm1.

During the **Indexing** process, a separate index is created for the source and target KGs. Classes from each KG are represented in documents that contain the concatenation of the class's instance names. The documents' contents are indexed using the standard Solr indexing process, including tokenisation, stemming, lemmatization, lower casing, and term-weighing. For our particular task an index is needed for NELL and DBpedia to perform the matching task. Thus, we run the following query to obtain all instance names for each DBpedia class:

```
SELECT ?name
```

[9] `https://lucene.apache.org/solr/`

```
WHERE{ ?entity a <http://dbpedia.org/ontology/%ClassName>.
       ?entity rdfs:label ?name.
       Filter (lang(?name)="en")}
```

After each query, a new document representing a DBpedia class is added and indexed in the designated DBpedia index. Similarly, an index was created for NELL which contained indexed documents of instance names parsed from the NELL facts dump.

Algorithm 1 Instance-Based Similarity Measure

Require:
source ← a list of classes in *Source* KG
target ← a list of classes in *Target* KG
for Class a_n in *source* **do**
  count = 1
  candidate = [ ]
  while count ≤ 30 **do**
    query ← a concatenation of 20 instance names of class a_n
    results ← search(*query*,*target*) in the *target* index
    for b_n in *results* **do**
      candidate.append(b_n)
    end for
    count++
  end while
  candidate_pairs ← Top three frequent classes in *candidate* paired with a_n
end for

To perform the matching process, NELL and DBpedia were treated as source and target respectively. Consequently, queries are generated by sampling instances names from NELL's classes. This process can be performed in the opposite direction; however, some of DBpedia's classes have missing instances. This implies that a query cannot be created from such empty classes. For example, classes such as `State`, `Zoo`, `Profession` are all leaf classes and supposed to be populated with individuals but the links between class's name and its instances are missing in the KG. A case in point is `California`[10] and `Florida`[11]: both are defined in the data with classes (i.e., rdf:type) other than `State`. This problem was encountered in 20 classes from the 138 classes selected from DBpedia. With DBpedia being the center of the LOD datasets in mind, many options can be explored in order to fulfill this gap. This includes using instances from *SKOS* concepts or another KG that already has an established mapping with DBpedia, such as WikiData or WebIsALOD. Nonetheless, we believe that performing a one-way search is sufficient for capturing all positive pairs for the annotation task.

In terms of the **Search** process, we aim to discover class pairs that share a significant number of overlapping instance names across two KGs. Our em-

[10] `http://dbpedia.org/page/California`
[11] `http://dbpedia.org/page/Florida`

pirical test on a smaller sample of the dataset showed that two key factors can directly impact the search (matching) result. The first one is the number of instance names to be used in the query string. Due to these KGs' instances being automatically extracted, and the large number of instances per class, using either a too-large or too-small number of instance names to create queries will result in no similar documents (classes) being retrieved or false positive pairs. The second factor impacting the search result is the number of searches (iterations) performed on each class to determine its equivalent class. Because of the restriction of the query length, concatenating the names of all class instances is not feasible. Moreover, by using a sample of instance names, different results can be retrieved depending on the sample. Our experiment has shown that we can obtain the maximum number of true positive pairs when concatenating 20 instances per query and performing 30 iterations per class.

To demonstrate, for a class a_n in NELL, a random 20 instances of that class are obtained and concatenated to form a query string. That query is then matched against all documents (classes) in the target index, i.e., DBpedia. Consequently, a list of classes whose instances overlap with those in the query are retrieved. For example, if the following results were retrieved when sampling instances from class $Airport_{nell}$ in the source KG:

Iteration 1 -> {$Airport_{dbp}$, $City_{dbp}$, $Port_{dbp}$}
Iteration 2 -> {$City_{dbp}$, $Port_{dbp}$, $Airport_{dbp}$}
Iteration 3 -> {$Airport_{dbp}$, $Port_{dbp}$}
Iteration n-1 -> {$Airport_{dbp}$, $City_{dbp}$}
Iteration n -> {$Airport_{dbp}$, $City_{dbp}$, $Street_{dbp}$}

By the end of the 30th iteration, we add three pairs of candidate alignments for class $Airport_{nell}$. Only the three most frequently retrieved classes among all iterations are added as positive pairs with a non-zero as similarity score. For the above example, the following pairs will be added: ($Airport_{nell}$,$Airport_{dbp}$), ($Airport_{nell}$,$City_{dbp}$), and ($Airport_{nell}$, $Port_{dbp}$). Notice that $Airport_{nell}$ is not matched to $Street_{dbp}$ as the latter only appeared once during the search process.

Combining Similarity Measures As our goal for this particular task is to discover potentially matching pairs to be annotated by human annotators, our aim is to ensure a high (if not full) recall, which was achieved by combining the two similarity measures. We applied the above mentioned similarity measures to the 18,492 class pairs obtained in the prior phase. Only pairs that obtained a similarity score higher than 0.4 by the String-based method or a non-zero value by the Instance-based method were considered for the annotation task. Following the above automated approach, we performed another manual screening to discover remaining equivalent classes from NELL and DBpedia that were not included in the potential pairs. By inspecting all pairs discarded by the filtering process we were able to identify and recover 8 pairs. A total of 596 pairs were created for the human annotation task.

3.4 Dataset Annotation

In order to create a gold-standard dataset of matching classes, we asked human annotators to determine the alignment for the previously discovered pairs, and then aggregate their interpretations by the majority votes, as human annotators can have different interpretation of correspondence. We have also performed a study of the inter-annotator agreement (IAA). The dataset was annotated by twenty research students and validated by two computer scientists. The participants were provided with guiding instructions to complete the task. Several labels were allowed to annotate pairs which are `a match`, `not a match`, `more general`, and `more specific`. The latter two options are often used in the ontology domain to label subsumption relation in ontologies. The reason we gave the annotators this option is that it can be possible in a few cases. For example, while DBpedia has two separate class for `State` and `Province`, NELL has one class named `StateOrProvince` which combines both.

Each participant annotated around 50 pairs on average. In order to observe (IAA), 400 random pairs are duplicated among 12 annotators such that each pair is annotated by 3 different annotators. The average IAA for this task was measured using Cohen's kappa based on a sample of the dataset and it was 0.83. The dataset was then validated by two experts. This was mainly to ensure that the subsumption relations were used properly. Therefore, a subsumption relation was only added to the dataset if there was an agreement by the experts. The gold standard mapping resulting from this annotation task is publicly available as two test cases[12]. The small test case includes a few instances per class, while the full test case contains the full A-box information for the included classes. The latter can be used to benchmark instance-based matching systems. The size of the gold standard is 129 equivalent class pairs with 24 non-trivial matches, i.e., not an exact matching string of class labels. Currently, the larger dataset in OAEI's KG track carries only 15 class matches, while the maximum number of non-trivial matches is 10. This makes the proposed dataset the largest domain-independent gold standard for matching KG classes. This gold standard is considered as a *partial gold standard* since some classes in both KGs have no equivalent class in the corresponding KG.

4 Evaluation

We evaluated the performance of the matching systems that participated in the KG track in OAEI 2019 event on the proposed gold standard. The Matching Evaluation Toolkit MELT [13] was used to perform this evaluation along with the SEALS client. The following systems were evaluated: POMAP++ [15], AML [9], FCAMap-KG [6], LogMap [14], LogMapLt, LogMapKG, LogMapBio, DOME [11], Wiktionary [18], and the string matcher used as a baseline for the KG track.

[12] `https://github.com/OmaimaFallatah/KG_GoldeStandard`

We evaluated the class alignments resulting from each matcher based on precision, recall, and f-measure. Results of the evaluation are shown in Table 4. Since the proposed gold standard is only a partial gold standard, and to avoid over-penalising systems that may discover reasonable matches that are not coded in our gold standard, we ignore any predicted matches if neither of the classes in that pair is present as a true positive pair with another class in our gold standard. As an example, for a class a_n we only consider the alignment *(a_n, b_n)* as a false positive, if the gold standard has a true positive pair containing either a_n or b_n but not both in the same pair.

Table 2. Performance of the KG track participants in OAEI on the proposed dataset compared to their performance on the OAEI KG track **starwars-swtor** benchmark

	Proposed Dataset			**OAEI KG benchmark**		
Matcher	**Precision**	**Recall**	**F1**	**Precision**	**Recall**	**F1**
POMAPP++	0.0	0.0	0.00	0.0	0.0	0.00
AML	1.00	0.61	0.75	1.00	0.87	0.93
FCAMap-KG	0.96	0.62	0.75	1.00	0.80	0.89
LogMap	0.98	0.79	0.88	1.00	0.80	0.89
LogMapKG	0.98	0.79	0.88	1.00	0.80	0.89
LogMapBio	0.98	0.79	0.88	1.00	0.80	0.89
LogMapLt	1.00	0.60	0.75	1.0	0.73	0.85
DOME	0.99	0.63	0.77	0.93	0.87	0.90
Wiktionary	0.99	0.79	0.88	1.00	0.87	0.93
KGbaselineLabel	1.0	0.61	0.76	1.00	0.80	0.89

As Table 4 shows, we have also evaluated the matchers on the **starwars-swtor** test case, which is the largest dataset in the track in terms of the size of class correspondences (which is 15). The best performing systems on the OAEI dataset in terms of recall are DOME, Wiktionary and AML; however, DOME and AML have obtained a lower recall (0.6) in our dataset, while Wiktionary is one of the best performing systems on our dataset. In contrast, the second to best performing matchers on the OAEI dataset, i.e., the LogMap family, obtained a recall of 0.79, which is the best recall on our gold standard. Nonetheless, 27 out of the 129 true positive pairs were not discovered by any matcher in the LogMap family. Among the evaluated matchers, LogMApKG and FCAMap-KG are the only systems that are particularly designed to match KGs. While the latter is the second best performing system in OAEI's 2019 KG track, particularly in matching classes, it has obtained a recall of 0.62 on our dataset. In terms of the precision on our dataset, the scores are fairly high since most systems were only able to discover trivial matches. However, the recall ranges between 0.6 and 0.79, which shows that the dataset contains class correspondences that are difficult to find. Hence, all systems need further improvements in order to map the classes of large and domain-independent KGs.

5 Discussion

From the results presented above, the following three patterns were observed. **First**, while current tools are able to produce high-quality results for well-formed ontologies, such techniques are not as well-performing when applied on KGs that lack textual descriptions. For instance, DOME is a matcher that trains a doc2vec model using all available metadata descriptions for ontologies. This can explain the matcher's low performance on our dataset as it requires a large amount of text. **Second**, many ontology matching systems utilizes structural knowledge available in well-structured ontologies such as disjoint axioms to refine their alignments [6]. Examples of systems that follow such an approach are AML and LogMap. However, as a result of the lack of schematic information in NELL, structural-based techniques can be difficult to apply in this case. **Third**, matching strategies used when two resources are from a specific domain setting are not applicable for domain-independent settings where classes contain information about real-world entities described with different terminologies. Therefore, in order to tackle the problem of KG matching, the need for specialized matching tools remains significant. Recently, many matching tools based on entity embedding are being proposed but only tested with domain-dependent datasets or in task-oriented settings, (e.g., [7,21]). Tailoring such methods for multi-domain KG matching and testing them on our gold standard can lead to deeper understanding and discovery in this domain.

6 Conclusion

In this paper, we developed the largest gold standard dataset for matching the classes of large KGs. Our gold standard is based on two highly influential KGs, and one of them is yet to be linked to the LOD. We evaluated several state-of-the-art matching tools on this dataset and showed that the task of matching large, domain-independent KGs remains very challenging. We argue that matching large, domain-independent and automatically constructed KGs has significant utility and therefore, future work should be devoted further into this area. We believe that our dataset and findings will foster research in this direction.

References

1. Algergawy, A., Faria, D., Ferrara, A., Fundulaki, I., Harrow, I., Hertling, S., Jiménez-Ruiz, E., Karam, N., Khiat, A., Lambrix, P., et al.: Results of the ontology alignment evaluation initiative 2019. In: CEUR Workshop Proceedings. pp. 46–85 (2019)
2. Anam, S., Kim, Y.S., Kang, B.H., Liu, Q.: Review of ontology matching approaches and challenges. International Journal of Computer Science and Network Solutions pp. 1–27 (2015)
3. Bizer, C., Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mende, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: DBpedia – A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia. Semantic Web pp. 1–5 (2012)

4. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E.R., Mitchell, T.M.: Toward an architecture for never-ending language learning. In: Twenty-Fourth AAAI Conference on AI (2010)
5. Cheatham, M., Hitzler, P.: String similarity metrics for ontology alignment. In: International semantic web conference. pp. 294–309 (2013)
6. Chen, G., Zhang, S.: Identifying mappings among knowledge graphs by formal concept analysis. In: OM@ ISWC. pp. 25–35 (2019)
7. Dhouib, M.T., Zucker, C.F., Tettamanzi, A.G.: An ontology alignment approach combining word embedding and the radius measure. In: International Conference on Semantic Systems. pp. 191–197 (2019)
8. Euzenat, J., Shvaiko, P.: Ontology matching: state of the art and future challenges. IEEE Transactions on Knowledge and Data Engineering (2013)
9. Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F., Couto, F.M.: The agreementmakerlight ontology matching system. In: OTM Confederated International Conferences "On the Move to Meaningful Internet Systems". pp. 527–541. Springer (2013)
10. Hertling, S., Paulheim, H.: Webisalod: providing hypernymy relations extracted from the web as linked open data. In: International Semantic Web Conference. pp. 111–119 (2017)
11. Hertling, S., Paulheim, H.: DOME results for OAEI 2018. CEUR Workshop Proceedings pp. 144–151 (2018)
12. Hertling, S., Paulheim, H.: The knowledge graph track at oaei. In: European Semantic Web Conference. pp. 343–359 (2020)
13. Hertling, S., Portisch, J., Paulheim, H.: Melt-matching evaluation toolkit. In: International Conference on Semantic Systems. pp. 231–245 (2019)
14. Jiménez-Ruiz, E.: Logmap family participation in the OAEI 2019. In: CEUR Workshop Proceedings. pp. 160–163 (2019)
15. Laadhar, A., Ghozzi, F., Megdiche Bousarsar, I., Ravat, F., Teste, O., Gargouri, F.: Pomap: An effective pairwise ontology matching system. In: 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management. pp. 161–168 (2017)
16. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady. pp. 707–710 (1966)
17. Paulheim, H.: Machine learning with and for semantic web knowledge graphs. In: Reasoning Web International Summer School. pp. 110–141. Springer (2018)
18. Portisch, J., Hladik, M., Paulheim, H.: Wiktionary matcher. In: CEUR Workshop Proceedings. pp. 181–188 (2019)
19. Ringler, D., Paulheim, H.: One knowledge graph to rule them all? In: Joint German/Austrian Conference on Artificial Intelligence. pp. 366–372 (2017)
20. Suchanek, F.M., Kasneci, G., Weikum, G.: Yago: a core of semantic knowledge. In: Proceedings of the 16th International Conference on World Wide Web. pp. 697–706 (2007)
21. Trisedya, B.D., Qi, J., Zhang, R.: Entity alignment between knowledge graphs using attribute embeddings. In: Proceedings of the AAAI Conference on AI. pp. 297–304 (2019)
22. Zhang, Z., Gentile, A.L., Blomqvist, E., Augenstein, I., Ciravegna, F.: An unsupervised data-driven method to discover equivalent relations in large Linked Datasets. Semantic Web pp. 197–223 (2017)

Applying edge-counting semantic similarities to Link Discovery: Scalability and Accuracy

Kleanthi Georgala[1,2], Mohamed Ahmed Sherif[2], Michael Röder[2], and Axel-Cyrille Ngonga Ngomo[1,2]

[1] Department of Computer Science, Paderborn University, Germany,
[2] Department of Computer Science, University of Leipzig, Germany
georgala@informatik.uni-leipzig.de
{mohamed.sherif,michael.roeder,axel.ngonga}@upb.de

Abstract. With the growth in number and variety of RDF datasets comes an increasing need for both scalable and accurate solutions to support link discovery at instance level within and across these datasets. In contrast to ontology matching, most linking frameworks rely solely on string similarities to this end. The limited use of semantic similarities when linking instances is partly due to the current literature stating that they (1) do not improve the F-measure of instance linking approaches and (2) are impractical to use because they lack time efficiency. We revisit the combination of string and semantic similarities for linking instances. Contrary to the literature, our results suggest that this combination can improve the F-measure achieved by instance linking systems when the combination of the measures is performed by a machine learning approach. To achieve this insight, we had to address the scalability of semantic similarities. We hence present a framework for the rapid computation of semantic similarities based on edge counting. This runtime improvement allowed us to run an evaluation of 5 benchmark datasets. Our results suggest that combining string and semantic similarities can improve the F-measure by up to 6% absolute.

1 Introduction

RDF knowledge graphs (KGs) are used in a plethora of applications [10], especially when published using the Linked Data paradigm. The provision of links[3] between such KGs is of central importance for numerous tasks such as federated queries [22] and question answering [25]. Popular solutions to linking instances (often called link discovery, short LD in the literature, see [12] for a survey) often implement specialized measures for particular datatypes (e.g., geospatial or temporal data). In all other cases, state-of-the-art LD frameworks such as SILK [1] and LIMES [14] rely on string similarities and machine learning to compute links between instances in RDF KGs. While the

 This work has been supported by the EU H2020 project KnowGraphs (GA no. 860801) as well as the BMVI projects LIMBO (GA no. 19F2029C) and OPAL (GA no. 19F2028A).

[3] The fourth principal of Linked Data, see http://www.w3.org/DesignIssues/LinkedData

use of string similarities has been shown to work well in a large number of papers (see, e.g., [12,4]), string similarities have the major drawback of not considering the semantics of the sequences of tokens they aim to compare. Hence, most string similarity measures return low scores for pairs of strings such as (`lift`, `elevator`), (`holiday`, `vacation`), (`headmaster`, `principal`) and (`aubergine`, `eggplant`), although they often stand for the same real-world concepts. Edge-counting semantic similarities (e.g., [26,9,20]) alleviate this problem by using a dictionary to compute a semantic distance between sequence of tokens within the need for an overlap. The synonymy between `aubergine` and `eggplant` would hence lead semantic similarity to assign the pair (`aubergine`, `eggplant`) a similarity score close to 1.

The use of semantic similarities has been paid little attention to in LD for at least two reasons: First, *semantic similarities scale poorly* and are thus impractical when used on large knowledge graphs.[4] Moreover, current works (e.g., [11]) suggest that they lead to no improvement in F-measure. The goal of this paper is hence twofold: (1) we present means to accelerate the computation of four popular bounded edge-counting semantic similarities. (2) We then combine string and semantic similarities using two state-of-the-art machine learning approaches for LD. Our results refute the current state of the art and suggest that semantic similarities can help achieve better results in LD.

2 Preliminaries

The formal framework underlying our preliminaries is derived from [23]. A KG K is a set of triples $(s, p, o) \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{I} \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L})$, where $\mathcal{I}$ is the set of all IRIs, $\mathcal{B}$ is the set of all RDF blank nodes and $\mathcal{L}$ is the set of all literals. LD frameworks aim to compute the set $M = \{(s, t) \in S \times T : R(s, t)\}$ where S and T are sets of RDF resources and R is a binary relation. Note that this setting generalizes what is often known as *entity matching* or *deduplication* [12], where the relation R must be `owl:sameAs`. Given that M is generally difficult to compute directly, declarative LD frameworks compute an approximation $M' \subseteq S \times T$ of M by executing a *link specification* (LS), which we define formally in the following. Let $\mathbb{M}$ be the set of all similarity functions. We define a similarity function $m \in \mathbb{M}$ as a function $m : S \times T \times \mathcal{P}^2 \rightarrow [0, 1]$, where $\mathcal{P}$ is the set of all properties, where $p_s, p_t \in \mathcal{P}$. We write $m(s, t, p_s, p_t)$ to signify the similarity of s and t w.r.t. their properties p_s resp. p_t. An *atomic LS* L is a pair $L = ((m(p_s, p_t), \theta)$, where $\theta \in [0, 1]$ is a similarity threshold. A *complex LS* L is a tuple $L = op(L_1, L_2)$ where two subspecification L_1 and L_2 are combined using the specification operator op. Here, we consider the binary operators union ($\sqcup$), intersection ($\sqcap$) and difference ($\setminus$).

The edge-counting semantic similarities are based on a lexical vocabulary. We define a *lexical vocabulary* as a directed acyclic graph (DAG) $G = (V, E)$, where:

- The set of vertices V is a set of concepts c_i, were each c_i stands for a set of synonyms. We denote $|V|$ with n_V.
- $E \subseteq V \times V$ is a set of directed edges $e_{jk} = (c_j, c_k)$. We denote $|E|$ with n_E.

[4] This general finding is supported by our evaluation results presented in Section 4.

- The *edge* e_{jk} stands for the hypernymy relation from a parent concept c_j to a child concept c_k. We write $c_j \rightarrow c_k$ and we say that c_j is a hypernym of c_k. We also define the hyponymy relation as a directed relation from a child concept c_k to a parent concept. We write $c_j \leftarrow c_k$ and we say that c_j is a hyponym of c_k. Hypernymy and hyponymy are transitive.
- The *root* r is the unique node of the dictionary that has no parent concept.
- A *leaf concept* c_i is a concept node without any children concepts.
- A concept is a *common subsumer* of c_1 and c_2 (denoted $cs(c_1, c_2)$) iff that concept is a hypernym of both c_1 and c_2.
- The *least common subsumer* (LSO) of c_1 and c_2 (denoted $lso(c_1, c_2)$) is "the most specific concept which is an ancestor of both c_1 and c_2" [26].
- We define the *directed path* from c_1 to c_2 via a common subsumer $cs(c_1, c_2)$ as: $path(c_1, c_2) = \{c_1 \leftarrow c_i \leftarrow \ldots \leftarrow cs(c_1, c_2) \rightarrow c_j \rightarrow \ldots \rightarrow c_2 : i, j, k \in \mathbb{N}, i, j, k \leq n_v\}$. Note that there can be multiple $path(c_1, c_2)$ between two concepts.
- $len(c_1, c_2)$ is *the length of the shortest* $path(c_1, c_2)$ between two concepts c_1 and c_2. Note that len defines a metric. Hence, it is symmetric and abides by the triangle inequality, i.e., $len(c_1, c_2) \leq len(c_1, c_3) + len(c_2, c_3)$ for any $(c_1, c_2, c_3) \in V^3$.
- We define $depth_m(c_i)$ as the length of the shortest path between r and c_i. Analogously, $depth_M(c_i)$ as the maximum $depth(c_i)$. We set $D = \max_{c \in V} depth_M(c)$.

Note that the following holds:

- $depth_m(r, c_i) = len(r, c_i)$
- $depth_m(lso(c_1, c_2)) \leq min(depth_m(c_1), depth_m(c_2))$
- $depth_M(lso(c_1, c_2)) \leq min(depth_M(c_1), depth_M(c_2))$
- (triangle inequality) $|len(r, c_1) - len(r, c_2)| \leq len(c_1, c_2) \Leftrightarrow |depth_m(c_1) - depth_m(c_2)| \leq len(c_1, c_2)$

The Shortest Path (SP) similarity [20] of two concepts c_1 and c_2 is defined as the length of their shortest path in comparison to the maximum distance ($2D$). We use the normalized formulation of SP, i.e.,

$$\text{SP}(c_1, c_2) = \frac{2D - len(c_1, c_2)}{2D}. \tag{1}$$

The Leacock and Chodorow metric (LCH) takes both the path between two concepts and the depth of the hierarchy into consideration [8]. We use the normalized formulation of LCH:

$$\text{LCH}_N(c_1, c_2) = \begin{cases} 1 & \text{if } c_1 = c_2 \\ \frac{-\log\left(\frac{len(c_1, c_2)}{2D}\right)}{\log(2D)} & \text{else.} \end{cases} \tag{2}$$

The normalized Wu Palmer (WP) similarity takes the path between two concepts and the depth of their LSO into consideration [26]:

$$\text{WP}(c_1, c_2) = \frac{2 \times depth_M(lso(c_1, c_2))}{2 \times depth_M(lso(c_1, c_2)) + N_1 + N_2} \tag{3}$$

where $N_1 = len(lso(c_1, c_2), c_1)$ and $N_2 = len(lso(c_1, c_2), c_2)$. The Li et al. metric (LI) is another take on using the path between two concepts and their LSO to define a similarity [9]:

$$\text{LI}(c_1, c_2) = e^{-\alpha len(c_1, c_2)} \frac{e^{\beta depth(lso(c_1, c_2))} - e^{-\beta depth(lso(c_1, c_2))}}{e^{\beta depth(lso(c_1, c_2))} + e^{-\beta depth(lso(c_1, c_2))}} \quad (4)$$

where $\text{LI}(c_1, c_2) \in (0, 1)$. We set $depth(lso(c_1, c_2)) = depth_M(lso(c_1, c_2))$, since the original specification does not state which $depth(lso(c_1, c_2))$ to use.

3 Approach

Fundamentally, hECATE aims to compute the set $M' = \{(s, t) \in S \times T : m(s, t, p_s, p_t) \geq \theta\}$, where m is an edge-counting similarity. To achieve this goal, the approach makes use of upper bounds which can be derived from the formulation of this family of measures. Take the SP similarity for example: For any two concepts c_1 and c_2, $\text{SP}(c_1, c_2) \geq \theta$ implies $len(c_1, c_2) \leq 2D(1 - \theta)$. Formally, this means that we can discard all comparisons of pairs (c_1, c_2) with $len(c_1, c_2) > 2D(1 - \theta)$ without compromising the computation of M'. Note that the computation of $len(c_1, c_2)$ can be carried out online or offline, which affects the total runtime of our approach as discussed in Section 4. As similar bounds can be derived for the other edge-counting measures, hECATE generalizes the computation of M' for edge-counting semantic similarities by using the following algorithm. Our approach takes (1) two sets of resources, S and T, (2) an atomic LS $L = ((m(p_s, p_t), \theta)$, where m is one of the four semantic similarities described in Section 2, and (3) a lexical vocabulary structured as DAG (VDAG) as input. Our goal is to compute the mapping $M' = [[L]]$ For each pair (s, t), hECATE retrieves and pre-processes the property values for p_s resp. p_t. The pre-processing consists of tokenizing and extracting all stop-words from the objects of the triples (s, p_s, o_s) and (t, p_t, o_t). In order to include a pair (s, t) in M', the algorithm compares each set of source tokens from o_s ($sTokens$) to each set of target tokens of o_t ($tTokens$). The pair of objects (o_s, o_t) with the highest similarity which abides by the bounds we derive for each measure is finally used to compute the similarity between s and t, and decides whether or not this pair should be added to M'. To do so, for each token $sToken \in sTokens$, we find the $tToken \in tTokens$ that is most similar. First, the algorithm checks if $sToken$ and $tToken$ have been compared before If the tokens are being compared for the first time the algorithm checks if the tokens are equal and assigns the value of 1 to $TTSim$. Otherwise, it calls the function `compare(sToken, tToken, VDAG)` that compares the corresponding sets of concepts obtained from the input `VDAG`. [5] Then, $TTSim$ is compared to the maximum token-to-token similarity and $maxTTSim$ is updated. The procedure continues until the highest similarity between the current $sToken$ and a $tToken$ is found or $maxTTSim$ is equal to 1. The algorithm aggregates the highest similarities $maxTTSim$ of all $sToken \in sTokens$ and calculates an average similarity. This is done for all pairs of ($sTokens$, $tTokens$) searching for the pair with the maximum similarity. If this $maxSimilarity > \theta$ the

[5] Note that our algorithm handles homonyms by considering that a token can be included in more than one concept.

pair (s, t) can be added to the final mapping M'. The key behind hECATE lies in the token comparison algorithm `compare(sToken, tToken, VDAG)` (Algorithms 1 and 3). For a pair of tokens `(sToken, tToken)`, we retrieve the set of concepts they belong to in the `VDAG`. If both sets of concepts are not empty, we compare each source `sCon` with each target concept `tCon` and define the maximum similarity of two tokens as the highest similarity of the corresponding concept pairs. To do so, we first retrieve the set of all hypernym paths of each concept to the root of the `VDAG` using the `getPaths(concept, VDAG)` algorithm. This algorithm traverses the `VDAG` by utilizing the hypernym relation. It starts from the *concept* node and explores all paths to the root node.For SP and LCH, we additionally retrieve the maximum depth D found in the `VDAG` and the `len(sCon, tCon)` before calculating the corresponding similarity as described in Equations 1 and 2 resp. For calculating `len(sCon, tCon)` our algorithm relies on the set of hypernym paths of the concepts (Algorithm 2). For each pair of hypernym paths hp_1 and hp_2 the two concepts have, the algorithm iterates over both paths simultaneously, from top to bottom, until they do not share a common node. Then, it proceeds in calculating the length of the newly found path, as the number of concepts that the two paths do not have in common. Finally, the minimum length that has been found is returned. For WP and LI, the comparison algorithm retrieves the depth of the LSO between `sCon` and `tCon` (`depth( lso(sCon, tCon))`, and N_1 and N_2 by calling the function `getLSO` (hps_1, hps_2) (Algorithm 4). This function utilizes the set of hypernym paths in a similar manner as the min length algorithm. For each combination of hypernym paths hp_1 and hp_2 of the concepts, the algorithm traverses them simultaneously searching for the last node they have in common. If this node is deeper than any other common node found so far or it has the same depth but the remaining paths are shorter, it is taken as new LSO. Accordingly, the remaining path lengths N_1 and N_2 are updated. Based on the deepest LSO and the derived values for $\texttt{depth}_M$ `(lso(sCon, tCon)`, N_1 and N_2, we proceed in calculating the corresponding similarity as described in Equations 3 and 4 resp.

Our first extension of hECATE is based on the idea of pre-computing and storing a set of values that are used often in our algorithm. For edge-counting similarities, these are the hypernym paths. Consequently, the extension hECATE-I of hECATE precomputes all hypernym paths for all concepts included in the `VDAG`, using the `getPaths( concept, VDAG)` function. Therefore, every time the `getPaths(concept, VDAG)` is invoked at runtime, hECATE-I retrieves the paths from an index. Our second extension of hECATE, hECATE-IF, combines hECATE-I with the idea of minimizing unnecessary comparison between concepts by filtering out pairs of source and target concepts that do not satisfy a condition for each semantic similarity. The filtering is performed inside `compare(sToken, tToken, VDAG)` for each pair of concepts `sCon` and `tCon`. Given a semantic similarity, if a pair of concepts satisfies the corresponding filtering condition, then the algorithm proceeds normally as described before. If the condition is not met the algorithm does not compute the similarity between the two concepts. For the SP similarity, two concepts will be considered for comparison, if the following holds:

$$\begin{aligned} \text{SP}(c_1, c_2) \geq \theta \Leftrightarrow \frac{2D - len(c_1, c_2)}{2D} \geq \theta \\ \Rightarrow |depth_m(c_1) - depth_m(c_2)| \leq 2D(1 - \theta) \end{aligned} \tag{5}$$

For the WP similarity, the following must hold:

Algorithm 1: $compare(sCon, tCon, VDAG)$ for SP or LCH

Input: source concept `sCon`, target concept `tCon`, and a vocabulary DAG `VDAG`
Output: a $similarity$ value

$D \leftarrow VDAG.getMaxDepth(sCon)$
$hps_1 \leftarrow getPaths(sCon, VDAG)$
$hps_2 \leftarrow getPaths(tCon, VDAG)$
$minLength \leftarrow getMinLength(hps_1, hps_2)$
Return $computeSimilarity(D, minLength)$

Algorithm 2: $getMinLength(hps_1, hps_2)$

Input: two sets of hypernym paths, hps_1 and hps_2
Output: $len(sCon, tCon)$

$size \leftarrow MAX\_VALUE$
foreach $hp_1 \in hps_1$ **do**
  foreach $hp_2 \in hps_2$ **do**
    $l_1 \leftarrow 0, l_2 \leftarrow 0$
    while $l_1 < hp_1.size() \wedge l_2 < hp_2.size() \wedge hp_1.get(l_1) == hp_2.get(l_2)$ **do**
      $l_1 \leftarrow l_1 + 1, l_2 \leftarrow l_2 + 1$
    $newSize \leftarrow hp_1.size() + hp_2.size() - 2l_1$
    if $newSize < size$ **then** $size \leftarrow newSize$;
Return $size$

Algorithm 3: $compare(sCon, tCon, VDAG)$ for WP or LI

Input: source concept `sCon`, target concept `tCon`, and a vocabulary DAG `VDAG`
Output: a $similarity$ value

$hps_1 \leftarrow getPaths(sCon, VDAG)$
$hps_2 \leftarrow getPaths(tCon, VDAG)$
$depth, N_1, N_2 \leftarrow getLSO(hps_1, hps_2)$
Return $computeSimilarity(N_1, N_2, depth)$

Algorithm 4: $getLSO(hps_1, hps_2)$

Input: two sets of hypernym paths, hps_1 and hps_2
Output: $depth_M(lso(sCon, tCon))$, N_1 and N_2

$dLSO \leftarrow 0, N_1 \leftarrow 0, N_2 \leftarrow 0$
foreach $hp_1 \in hps_1$ **do**
  foreach $hp_2 \in hps_2$ **do**
    $l_1 \leftarrow 0, l_2 \leftarrow 0$
    while $l_1 < hp_1.size() \wedge l_2 < hp_2.size() \wedge hp_1.get(l_1) == hp_2.get(l_2)$ **do**
      $l_1 \leftarrow l_1 + 1, l_2 \leftarrow l_2 + 1$
    $newSize \leftarrow hp_1.size() + hp_2.size() - 2l_1$
    $oldSize \leftarrow N_1 + N_2$
    if *condition is met* **then**
      $dLSO \leftarrow l_1, N_1 \leftarrow hp_1.size() - l_1$
      $N_2 \leftarrow hp_2.size() - l_2$
Return $dLSO, N_1, N_2$

$$
\begin{aligned}
WU(c_1, c_2) \geq \theta &\Leftrightarrow \frac{2depth_M(lso(c_1, c_2))}{2depth_M(lso(c_1, c_2)) + N_1 + N_2} \geq \theta \\
&\Leftrightarrow 2depth_M(lso(c_1, c_2)) \geq \theta(N_1 + N_2) + 2\theta depth_M(lso(c_1, c_2)) \\
&\Leftrightarrow N_1 + N_2 \leq \frac{2depth_M(lso(c_1, c_2))(1 - \theta)}{\theta} \\
&\Rightarrow N_1 + N_2 \leq \frac{2\min(depth_M(c_1), depth_M(c_2))(1 - \theta)}{\theta}
\end{aligned}
\tag{6}
$$

Based on the triangle inequality and Section 2, Equation 6 can be written as:

$$\begin{aligned} len(c_1, c_2) \leq \frac{2\min(depth_M(c_1), depth_M(c_2))(1-\theta)}{\theta} \Rightarrow \\ |depth_m(c_1) - depth_m(c_2)| \leq \frac{2min(depth_M(c_1), depth_M(c_2))(1-\theta)}{\theta} \end{aligned} \tag{7}$$

For the LCH similarity, two concepts will be considered for comparison, iff:

$$\begin{aligned} \text{LCH}(c_1, c_2) \geq \theta \Leftrightarrow \frac{-log\frac{len(c_1,c_2)}{2D}}{log(2D)} \geq \theta \Leftrightarrow \frac{log(2D) - log(len(c_1, c_2))}{log(2D)} \geq \theta \Leftrightarrow \\ 1 - \frac{log(len(c_1, c_2))}{log(2D)} \geq \theta \Leftrightarrow log(len(c_1, c_2)) \leq log(2D)(1-\theta) \Leftrightarrow \\ len(c_1, c_2) \leq 2^{log(2D)(1-\theta)} \Rightarrow |depth_m(c_1) - depth_m(c_2)| \leq 2^{log(2D)(1-\theta)} \end{aligned} \tag{8}$$

When considering the LI similarity, we make the following variable replacements for the sake of legibility: $x = depth_M(lso(c_1, c_2))$, $y = min(depth_M(c_1), depth_M(c_2))$ and $z = len(c_1, c_2)$. Then, two concepts will be considered for comparison, iff:

$$\begin{aligned} \text{LI}(c_1, c_2) \geq \theta \Leftrightarrow e^{-\alpha z}\frac{e^{\beta x} - e^{-\beta x}}{e^{\beta x} + e^{-\beta x}} \geq \theta \Leftrightarrow e^{\alpha z} \leq \frac{e^{\beta x} - e^{-\beta x}}{(e^{\beta x} + e^{-\beta x})\theta} \Leftrightarrow e^{\alpha z} \leq \frac{\frac{(e^{2\beta x}-1)}{e^{\beta x}}}{\frac{(e^{2\beta x}+1)}{e^{\beta x}}\theta} \Leftrightarrow \\ e^{\alpha z} \leq \frac{(e^{2\beta x}-1)}{(e^{2\beta x}+1)\theta} \Rightarrow e^{\alpha z} \leq \frac{(e^{2\beta y}-1)}{(e^{2\beta y}+1)\theta} \Leftrightarrow \alpha z \leq ln(e^{2\beta y}-1) - ln\theta - ln(e^{2\beta y}+1) \Leftrightarrow \\ |depth_m(c_1) - depth_m(c_2)| \leq \frac{ln(e^{2\beta y}-1) - ln\theta - ln(e^{2\beta y}+1)}{\alpha} \end{aligned} \tag{9}$$

Based on Equations 5, 7, 8 and 9, each filtering condition requires the knowledge of $depth_m(sCon)$, $depth_M(sCon)$, $depth_m(tCon)$ and $depth_M(tCon)$. Hence, we further extend the index hECATE-IF relies on by precomputing $depth_m(c_i)$ and $depth_M(c_i)$ for every concept c_i.

4 Evaluation

Our evaluation addresses the following three research questions: Q_1. How do our strategies for improving the runtime of semantic similarities compare to each other w.r.t. runtime?, Q_2. How do the different edge-counting semantic similarities compare w.r.t. runtime?, and Q_3. Can semantic similarities improve the F-measure of LD systems?

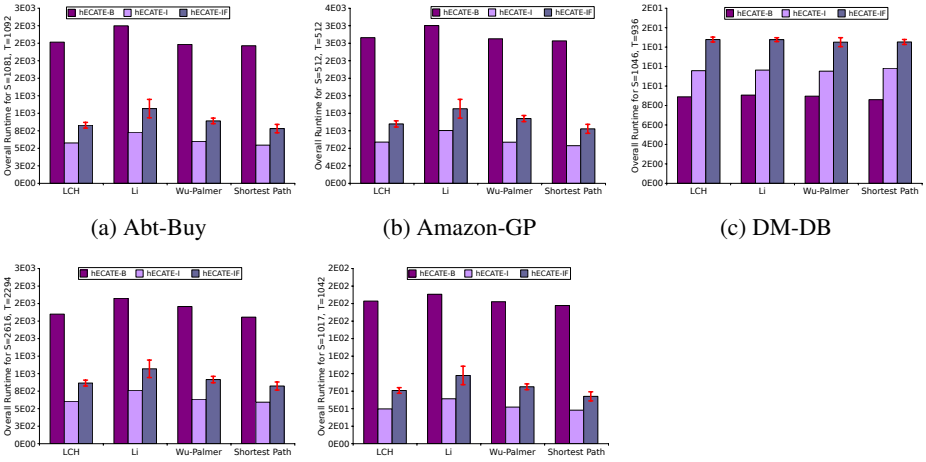
We evaluate our approach against five benchmark data sets: Abt-Buy, Amazon-GP and DBLP-ACM described in [7], DailyMed-Drugbank (dubbed DM-DB) and Movies described in [16]. We use WordNet[6] as a DAG. To address Q_1 and Q_2, we conduct a set of experiments using the basic hECATE algorithm (dubbed hECATE-B) as a baseline as well as hECATE-I and hECATE-IF. For an easier comparison, all methods are implemented in the LD framework LIMES [14]. For hECATE-B and hECATE-I, we create one atomic LS for each semantic similarity, where m iss the name of the edge-counting similarity, $\theta = 0.1$. We use the `'description'` as the source and target properties for *Abt-Buy* and *Amazon-GP* datasets, `'title'` for the *DBLP-Scholar* and

[6] `https://wordnet.princeton.edu/`

Movies datasets and `'name'` for the *DM-DB* dataset. For hECATE-IF, we use the same values for m, p_s and p_t as before, but θ is derived from the interval $[0.1, 1]$ with an increment step of 0.1, since the θ is given as a parameter to the filtering functions. For each dataset, we perform the aforementioned LSs against 2^v instances from the source and target datasets. We start with $v = 2$ and increment v until all instances are covered (e.g., the maximal value of v is 9 for the Amazon-Google dataset). We define a maximum runtime for each LS of $2\,hrs$. Each experiment is executed 3 times and we present the average values.

As explained in Section 1, the second goal of this work is to evaluate edge-counting semantic similarities in LD in terms of accuracy. Consequently, for Q_3, we use the hECATE extension with the best runtime performance based on the results of Q_1 and executed a set of experiments using 2 machine learning (ML) algorithms: WOMBAT [23] and DRAGON [19]. We choose these two approaches because (1) they achieve state-of-the-art performance while being deterministic, (2) they are open-source, meaning our experiments can be easily reproduced and (3) they are able to generate complex link specifications with any arbitrary number of measures. We perform a 10-fold cross validation by allowing WOMBAT and DRAGON to use only string similarities (StrSim), only semantic similarities (SmtSim) and a combination of both (StrSmtSim) as input. We use the `levenshtein`, `cosine` and `qgrams` similarity measures for strings implemented in LIMES [14]. For each dataset, we use all properties apart from those that corresponded to numeric values. WOMBAT is configured as presented in [23] and DRAGON is configured as presented in [19]. We use two termination criteria for WOMBAT: Either a LS with F-measure of 1 is found or a maximal depth of refinement of 10 is reached. For the string similarities, WOMBAT produced LSs with a minimum θ value of 0.4 and for the semantic similarities, the minimum θ value is set to 0.7. DRAGON terminates either when no new nodes are found or when the height of the decision tree reached the maximum of 3. Additionally, we compare the achieved F1 scores with scores for EAGLE [15], EUCLID [13], J48 [5] reported by [19], a Multilayer Perceptron classifier reported by [24] and the *Pessimistic* as well as *Re-weighted* versions of the work presented at [6].

As expected, Figure 1 shows that hECATE-B has the highest runtimes compared to hECATE-I and hECATE-IF in all datasets, except DM-DB. This supports the claim that semantic similarities typically scale poorly. The results show that both extensions improve the runtime of all semantic similarities, making them more amenable for LD and scalable for larger datasets. Precisely, LCH's, WP's and SP's runtimes improve by 71% and 57% on average when hECATE-I and hECATE-IF strategies are used resp. LI has the least improvement by 65% and 50%. Comparing the two extensions, in all datasets and for all semantic similarities, hECATE-I outperforms hECATE-IF by 30% on average. A detailed analysis of the runtimes shows that even though hECATE-IF reduces the number of comparisons between semantically different concepts and thus the comparison time, the additional runtime cost of filtering creates an overhead that results in a worse total execution time than hECATE-I (Table 1). Regarding the DM-DB dataset, the only property for both source and target datasets, *name*, consists of only one value, which corresponds to the official name of a drug. That value can only be associated with one concept. As a result, introducing an indexing and/or filtering technique



(a) Abt-Buy (b) Amazon-GP (c) DM-DB

(d) DBLP-ACM (e) Movies

Fig. 1: Average runtime in seconds of hECATE-B, hECATE-I and hECATE-IF on all datasets. For hECATE-IF, the standard deviation among different θ values is added.

produces an unnecessary overhead. Overall, Q_1 can be answered with hECATE-I being the most efficient approach.

To answer Q_2 we compare the runtimes of the single semantic similarities revealing that LI has the worst runtime (see Figure 1). For the Movies dataset, we notice that hECATE-I requires 100K more token comparisons for LI compared to the other similarities (Table 1). The better runtime of the other similarities is caused by a condition inside our algorithm which stops as soon as two tokens/concepts have a similarity of 1. In contrast to the other similarities, $\text{LI}(c_1, c_2) \in (0, 1)$, i.e., it can never be 1. However, based on Table 1, LI's runtime shows a great improvement as the values of θ increase in relation to the other metrics. This justifies the fact that the runtimes for LI have the highest standard deviation, whereas SP, LCH and WP are less influenced by the differ-

Table 1: Number of concept comparisons performed by hECATE-I and hECATE-IF for the Movies dataset. The numbers for hECATE-B are the same as for hECATE-I.

Threshold		0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
SP	hECATE-I	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M
	hECATE-IF	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	**61.0M**	**51.4M**	**10.3M**
WP	hECATE-I	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M
	hECATE-IF	61.8M	61.8M	61.8M	**61.7M**	**61.5M**	**60.3M**	**56.7M**	**44.7M**	**27.8M**	**10.3M**
LCH	hECATE-I	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M	61.8M
	hECATE-IF	61.8M	61.8M	61.8M	**61.4M**	**60.4M**	**56.5M**	**42.4M**	**42.4M**	**28.5M**	**28.5M**
LI	hECATE-I	61.9M	61.9M	61.9M	61.9M	61.9M	61.9M	61.9M	61.9M	61.9M	61.9M
	hECATE-IF	61.9M	**61.4M**	**59.9M**	**56.6M**	**51.5M**	**42.1M**	**28.3M**	**28.2M**	**10.0M**	**00.0M**

Table 2: Average F-measure achieved by WOMBAT, DRAGON, EUCLID, EAGLE, J48 and Multilayer Perception within a 10-fold cross validation. The semantic similarities use the hECATE-I strategy.

Algorithm	WOMBAT			DRAGON			EUCLID	EAGLE	J48	Perceptron
Similarities	StrSim	SmtSim	StrSmtSim	StrSim	SmtSim	StrSmtSim	StrSim	StrSim	StrSim	StrSim
Abt-Buy	0.65	0.65	**0.66**	0.51	0.02	0.10	0.00	0.56	0.43	0.43
Amazon-GP	0.71	0.60	**0.77**	0.64	0.06	0.05	0.71	0.73	0.41	0.36
DBLP-ACM	0.97	0.74	0.97	0.93	0.81	0.93	**0.98**	**0.98**	0.77	0.97
DM-DB	0.94	0.71	0.97	0.89	0.65	0.89	**1.00**	**1.00**	0.94	-
Movies	1.00	0.73	**1.00**	0.93	0.80	0.93	0.98	0.99	0.84	-

Table 3: Maximum F-measure achieved by WOMBAT, DRAGON, Pessimistic and Re-weighted using 2% of the data for training over 7 iterations [6]. The semantic similarities use the hECATE-I strategy.

Algorithm	WOMBAT			DRAGON			Pessimistic	Re-weighted
Similarities	StrSim	SmtSim	StrSmtSim	StrSim	SmtSim	StrSmtSim	StrSim	StrSim
Abt-Buy	0.35	**0.39**	0.34	0.24	0.10	0.24	0.36	0.37
Amazon-GP	**0.53**	0.33	0.43	0.45	0.13	0.35	0.39	0.43
DBLP-ACM	0.91	0.55	0.91	0.90	0.66	0.90	0.93	**0.95**
DM-DB	0.94	0.71	**0.97**	0.94	0.71	0.96	-	-
Movies	0.97	0.33	**0.97**	0.96	0.33	0.96	-	-

ent values of θ. The answer for Q_2 is that for all hECATE strategies, SP is the fastest similarity, whereas LI is the slowest.

To answer Q_3, we add the 4 edge-counting measures LI, WP, SP, and LCH to the state-of-the-art algorithms WOMBAT [23] and DRAGON [19]. We evaluate their performance with and without string similarities using a ten-fold cross validation. Table 2 shows the results of our experiments with these machine-learning algorithms. In the 6 right most columns of Table 2, we report the F1 score of the string-based LD algorithms. While the performance of DRAGON remained the same or even worsened for 3 of the 5 datasets, adding semantic similarities to the WOMBAT algorithm improved its overall performance for 3 datasets by up to 6% F-measure absolute. As expected, this effect is most pronounced in datasets which rely on long textual descriptions such as Amazon-GP. A look into the specifications learned by WOMBAT suggests that this effect is due to the approach combining semantic and string similarities using operators such as $\sqcup$ and learning the correct threshold for each of these measures. The improvement on the DM-DB datasets is achieved using the $\setminus$ operator, not allowing semantically similar concepts to be matched together. This refutes current results (see [11] where the same similarities have been used) and suggests that the refinement operators can combine semantic and string similarities in a way that improves the F-measure. For enabling a comparison with [6], we used the same configuration setting and report the maximum F-measure in Table 3. It can be seen that WOMBAT outperforms the Pessimistic and Re-weighted methods on the majority of the datasets.

5 Related Work

We give a brief overview of linking approaches which use semantic similarities. An exhaustive list of frameworks can be found in [12]. Over the past few years, semantic similarities were used in ontology matching (OM) [21]. In this context, concepts in two ontologies O_1 and O_2 are often matched based on a third ontology, e.g., WordNet. This ontology can be viewed as a background knowledge source or a mediating ontology [2]. Frameworks such as *AgreementMaker* [3], *Zhishi.links* [18] and *RuleMiner* [17] utilize semantic similarities in this way to improve structural matching on the ontology level. While these enhancements have a positive effect on their instance level matching, to the best of our knowledge no instance linking tool has used semantic similarities directly and shown an improvement of the overall linking results. [11] compare the effect of a predefined set of combinations of string and semantic similarities for label comparison and suggest that semantic similarities do not improve the F-measure of the instance matching task. Our results suggest the contrary by showing that dataset-specific combinations of measures actually can achieve a better performance.

6 Conclusions and Future Work

To study the effect of semantic similarities on LD, we presented hECATE, a generic framework for improving the runtime of edge-counting semantic similarities. Our evaluation of the framework shows that there is still a lot of potential in improving the runtime of semantic similarities for LD. We used hECATE to evaluate the performance of string similarities in LD on five datasets. Our evaluation shows that combining semantic similarities with string similarities can indeed increase the F-measure achieved by LD algorithms. This result is of central importance as it goes against current assumptions. The reason why we are indeed able to use semantic similarities for improving the F-measure of LD in some cases lies in the refinement operator employed by WOMBAT. In future works, we will investigate means that will allow improving the runtimes of semantic similarities, extend our works beyond edge-counting similarities and aim to classify datasets w.r.t. how suitable they are for semantic similarities.

References

1. Bizer, C., Volz, J., Kobilarov, G., Gaedke, M.: Silk - A Link Discovery Framework for the Web of Data. In: 18th International World Wide Web Conference (April 2009)
2. Cross, V., Silwal, P., Morell, D.: Using a reference ontology with semantic similarity in ontology alignment. In: Proceedings of the 3rd ICBO (2012)
3. Cruz, I.F., Antonelli, F.P., Stroe, C.: Agreementmaker: Efficient matching for large real-world schemas and ontologies. PVLDB **2**, 1586–1589 (2009)
4. Euzenat, J., Ferrara, A., Meilicke, C., Nikolov, A., Pane, J., Scharffe, F., Shvaiko, P., Stuckenschmidt, H., Šváb-Zazamal, O., Svátek, V., et al.: Results of the ontology alignment evaluation initiative 2010. Tech. rep., University of Trento (2011)
5. Holmes, G., Donkin, A., Witten, I.H.: Weka: a machine learning workbench. In: Proceedings of ANZIIS '94. pp. 357–361 (Nov 1994)

6. Kejriwal, M., Miranker, D.P.: Semi-supervised instance matching using boosted classifiers. In: The Semantic Web. Latest Advances and New Domains. pp. 388–402. Springer International Publishing (2015)
7. Köpcke, H., Thor, A., Rahm, E.: Evaluation of Entity Resolution Approaches on Real-world Match Problems. Proc. VLDB Endow. **3**(1-2), 484–493 (Sep 2010)
8. Leacock, C., Chodorow, M.: Combining local context and wordnet similarity for word sense identification (01 1998)
9. Li, Y., Bandar, Z.A., McLean, D.: An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans. on Knowl. and Data Eng. **15**(4), 871–882 (Jul 2003)
10. Malyshev, S., Krötzsch, M., González, L., Gonsior, J., Bielefeldt, A.: Getting the most out of wikidata: Semantic technology usage in wikipedia's knowledge graph. In: International Semantic Web Conference. pp. 376–394 (2018)
11. McCrae, J.P., Buitelaar, P.: Linking Datasets Using Semantic Textual Similarity. CYBERNETICS AND INFORMATION TECHNOLOGIES **18**(1), 109–123 (2018)
12. Nentwig, M., Hartung, M., Ngonga Ngomo, A.C., Rahm, E.: A survey of current link discovery frameworks. Semantic Web pp. 1–18 (2015)
13. Ngomo, A.C.N., Lyko, K.: Unsupervised learning of link specifications: deterministic vs. non-deterministic. In: OM (2013)
14. Ngonga Ngomo, A.C.: On Link Discovery using a Hybrid Approach. Journal on Data Semantics **1**(4), 203–217 (2012)
15. Ngonga Ngomo, A.C., Lyko, K.: Eagle: Efficient active learning of link specifications using genetic programming. In: The Semantic Web: Research and Applications. pp. 149–163. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
16. Ngonga Ngomo, A.C., Lyko, K.: Unsupervised learning of link specifications: deterministic vs. non-deterministic. In: Proceedings of the Ontology Matching Workshop (2013)
17. Niu, X., Rong, S., Wang, H., Yu, Y.: An effective rule miner for instance matching in a web of data. In: Proceedings of the 21st ACM International Conference on Information and Knowledge Management. pp. 1085–1094. CIKM '12, ACM, New York, NY, USA (2012)
18. Niu, X., Rong, S., Zhang, Y., Wang, H.: Zhishi.links results for OAEI 2011. Ontology Matching p. 220 (2011)
19. Obraczka, D., Ngomo, A.C.N.: Dragon: Decision tree learning for link discovery. In: 19TH International Conference On Web Engineering. Springer (2019)
20. Rada, R., Mili, H., Bicknell, E., Blettner, M.: Development and application of a metric on semantic nets. IEEE Trans. Systems, Man, and Cybernetics **19**, 17–30 (1989)
21. Rodríguez, M.A., Egenhofer, M.J.: Determining semantic similarity among entity classes from different ontologies. IEEE Trans. on Knowl. and Data Eng. **15**(2), 442–456 (2003)
22. Saleem, M., Ali, M.I., Verborgh, R., Ngonga Ngomo, A.C.: Federated query processing over linked data. In: Tutorial at ISWC (2015)
23. Sherif, M., Ngonga Ngomo, A.C., Lehmann, J.: WOMBAT - A Generalization Approach for Automatic Link Discovery. In: 14th Extended Semantic Web Conference, Portorož, Slovenia, 28th May - 1st June 2017 (2017)
24. Soru, T., Ngomo, A.C.N.: A comparison of supervised learning classifiers for link discovery. In: Proceedings of the 10th Intern. Conf. on Semantic Systems. pp. 41–44. ACM (2014)
25. Usbeck, R., Ngonga Ngomo, A.C., Haarmann, B., Krithara, A., Röder, M., Napolitano, G.: 7th open challenge on question answering over linked data (QALD-7). In: Semantic Web Evaluation Challenge. pp. 59–69. Springer International Publishing (2017)
26. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32Nd Annual Meeting on Association for Computational Linguistics. pp. 133–138. ACL '94, Association for Computational Linguistics, Stroudsburg, PA, USA (1994)

LIGON – Link Discovery with Noisy Oracles

Mohamed Ahmed Sherif[1,2], Kevin Dreßler[2], and Axel-Cyrille Ngonga Ngomo[1,2]

[1] Paderborn University, Data Science Group, Technologiepark 6, 33100 Paderborn, Germany
E-mail: {firstname.lastname}@upb.de
[2] Department of Computer Science, University of Leipzig, 04109 Leipzig, Germany
E-mail: {lastname}@informatik.uni-leipzig.de

Abstract. Link discovery plays a key role in the integration and use of data across RDF knowledge graphs. Active learning approaches are a common family of solutions to address the problem of learning how to compute links from users. So far, only active learning from perfect oracles has been considered in the literature. However, real oracles are often far from perfect (e.g., in crowdsourcing). We hence study the problem of learning how to compute links across knowledge graphs from noisy oracles, i.e., oracles that are not guaranteed to return correct classification results. We present a novel approach for link discovery based on a probabilistic model, with which we estimate the joint odds of the oracles' guesses. We combine this approach with an iterative learning approach based on refinements. The resulting method, LIGON, is evaluated on 11 benchmark datasets. Our results suggest that LIGON achieves more than 95% of the F-measure achieved by state-of-the-art algorithms trained with a perfect oracle.

1 Introduction

The provision of links between knowledge graphs in RDF[3] is of central importance for numerous tasks on the Semantic Web, including federated queries, question answering and data fusion. While links can be created manually for small knowledge bases, the sheer size and number of knowledge bases commonly used in modern applications (e.g., DBpedia with more than 3×10^6 resources) demands the use of automated link discovery mechanisms. In this work, we focus on active learning for link discovery. State-of-the-art approaches that rely on active learning [3, 12, 8] assume that the oracle they rely upon is *perfect*. Formally, this means that given an oracle ω, the probability of the oracle returning a wrong result (i.e., returning `false` when an example is to be classified as `true`) is exactly 0. While these approaches show pertinent results in evaluation scenarios, within which the need for a perfect oracle can be fulfilled, this need is difficult if

[3] See https://www.w3.org/RDF/.

not impossible to uphold in real-world settings (e.g., when crowdsourcing training data). No previous work has addressed link discovery based on oracles that are not perfect.

We address this research gap by presenting a novel approach for learning link specifications (LS) from *noisy oracles*, i.e., oracles that are not guaranteed to return correct classifications. This approach is motivated by the problem of learning LS using crowdsourcing. Previous works have shown that agents in real crowdsourcing scenarios are often not fully reliable (e.g., [19]). We model these agents as noisy oracles, which provides erroneous answers to questions with a fixed probability. We address the problem of learning from such oracles by using a probabilistic model, which approximates the odds of the answer of a set of oracles being correct. Our approach, dubbed LIGON, assumes that the underlying oracles are *independent*, i.e., that the probability distributions underlying oracles are pairwise independent. Moreover, we assume that the oracles have a *static behavior*, i.e., that the probability of them generating correct/incorrect answers is constant over time.

The contributions of this paper are as follows: (1) We present a formalization of the problem of learning LS from noisy oracles. We derive a probabilistic model for learning from such oracles. (2) We develop the first learning algorithm dedicated to learning LS from noisy data. The approach combines iterative operators for LS with an entropy-based approach for selecting most informative training examples. In addition, it uses cumulative evidence to approximate the probability distribution underlying the noisy oracles that provide it with training data. Finally, (3) we present a thorough evaluation of LIGON and show that it is robust against noise, scales well and converges with 10 learning iterations to more than 95% of the average F-measure achieved by WOMBAT—a state-of-the-art approach for learning LS—provided with a perfect oracle.

2 Preliminaries

Knowledge graphs (also called knowledge bases) in RDF are defined as sets of triples $K \subseteq (\mathcal{R} \cup \mathcal{B}) \times \mathcal{P} \times (\mathcal{R} \cup \mathcal{B} \cup \mathcal{L})$, where $\mathcal{R}$ is the set of all resources, i.e., of all objects in the domain of discourse (e.g., persons and publications); $\mathcal{P} \subseteq \mathcal{R}$ is the set of all predicates, i.e., of binary relations (e.g., author); $\mathcal{B}$ is the set of all blank nodes, which basically stand for resources whose existence is known but whose identity is not relevant to the model and $\mathcal{L}$ is the set of all literals, i.e., of values associated to datatypes (e.g., integers).[4] The elements of K are referred to as *facts* or *triples*. We call the elements of $\mathcal{R}$ *entities* or *resources*.

The *link discovery* task on RDF knowledge graphs is defined as follows: Let S and T be two sets of resources, i.e., $S \subseteq \mathcal{R}$ and $T \subseteq \mathcal{R}$. Moreover, let $r \in \mathcal{P}$ be a predicate. The aim of link discovery is to compute the set $M = \{(s,t) \in S \times T : r(s,t)\}$. We call M a mapping. In many cases, M cannot be computed directly and is thus approximated by a mapping M'. To find the set M', declarative

[4] See https://www.w3.org/RDF/ for more details.

Fig. 1: Complex LS example. The filter nodes are rectangles while the operator nodes are circles.

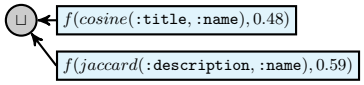


Table 1: Link Specification Syntax and Semantics

LS	$[[LS]]_M$
$f(m, \theta)$	$\{(s,t)\|(s,t) \in M \wedge m(s,t) \geq \theta\}$
$L_1 \sqcap L_2$	$\{(s,t)\|(s,t) \in [[L_1]]_M \wedge (s,t) \in [[L_2]]_M\}$
$L_1 \sqcup L_2$	$\{(s,t)\|(s,t) \in [[L_1]]_M \vee (s,t) \in [[L_2]]_M\}$
$L_1 \backslash L_2$	$\{(s,t)\|(s,t) \in [[L_1]]_M \wedge (s,t) \notin [[L_2]]_M\}$

link discovery frameworks rely on *link specifications* (LS), which describe the conditions under which $r(s,t)$ can be assumed to hold for a pair $(s,t) \in S \times T$. Several formal models have been used for describing LS in previous works [8]. We adopt a formal approach derived from [17] and first describe the syntax and then the semantics of LS.

LS consist of two types of atomic components: *similarity measures* m, which allow the comparing of property values of input resources and *operators* op, which can be used to combine LS to more complex LS. Without loss of generality, we define a similarity measure m as a function $m : S \times P \times T \times P \to [0,1]$. An example of a similarity measure is the edit similarity dubbed `edit`[5] which allows computing the similarity of a pair $(s,t) \in S \times T$ w.r.t. the values of a pair of properties (p_s, p_t) for s resp. t. An *atomic LS* is a pair (m, θ). A *complex LS* is the result of combining two LS L_1 and L_2 through an *operator* that allows merging the results of L_1 and L_2. Here, we use the operators $\sqcap$, $\sqcup$ and $\backslash$ as they are complete w.r.t. the Boolean algebra and frequently used to define LS. An example of a complex LS is given in Figure 1.

We define the semantics $[[L]]_\mu$ of a LS L w.r.t. a mapping μ as given in Table 1. The mapping $[[L]]$ of a LS L w.r.t. $S \times T$ contains the link candidates generated by L. A LS L is *subsumed* by L', denoted by $L \sqsubseteq L'$, if for all mappings μ, we have $[[L]]_\mu \subseteq [[L']]_\mu$. Two LS are *equivalent*, denoted by $L \equiv L'$ iff $L \sqsubseteq L'$ and $L' \sqsubseteq L$. Subsumption ($\sqsubseteq$) is a partial order over the set of LS, denoted $\mathcal{L}$.

3 Noisy Oracles

We model *oracles* Ω for r as black boxes with a characteristic function $\omega : S \times T \to \{\texttt{true}, \texttt{false}\}$. The characteristic function ω_i of the oracle Ω_i returns `true` iff the oracle Ω_i assumes that $r(s,t)$ holds. Otherwise, it returns `false`. For ease of notation, we define $LC = S \times T$ and call the elements of LC *link candidates*. For $l \in LC$, we write $l \equiv \top$ to signify that $r(l)$ holds, i.e., $r(s,t)$ is true for $l = (s,t)$. Otherwise, we write $l \equiv \bot$. We now assume a learning situation typical for crowdsourcing, where n oracles are presented with a link candidate l and asked whether $l \equiv \top$ holds. We can describe each oracle Ω_i by the following four probabilities: ① $p(\omega_i(l) = \texttt{true} | l \equiv \top)$, i.e., the probability of the oracle Ω_i generating true positives. This value is exactly 1 for a perfect

[5] We define the edit similarity of two strings s and t as $(1 + lev(s,t))^{-1}$, where lev is the Levenshtein distance.

oracle. ② $p(\omega_i(l) = \texttt{false} | l \equiv \top)$, the probability of false negatives (0 for a perfect oracle). ③ $p(\omega_i(l) = \texttt{true} | l \equiv \bot)$, i.e., the probability of false positives, (0 for a perfect oracle), and ④ $p(\omega_i(l) = \texttt{false} | l \equiv \bot)$, the probability of true negatives (1 for a perfect oracle). Given that $p(A|B) + p(\neg A|B) = 1$, the sum of the first two and last two probabilities is always 1.

Example 1. A noisy oracle can have the following description: $p(\omega_i(l)=\texttt{true}|l\equiv\top)=0.7, p(\omega_i(l)=\texttt{true}|l\equiv\bot)=0.5, p(\omega_i(l)=\texttt{false}|l\equiv\top)=0.3, p(\omega_i(l)=\texttt{false}|l\equiv\bot)=0.5$.

For compactness, we use the following vector notation in the rest of the formal model: $\overrightarrow{\omega}$ refers to the vector of characteristic functions over all oracles. We write $\overrightarrow{\omega}(l) = \overrightarrow{x}$ to signify that the ith oracle returned the x_i for the link candidate l. Let us assume that the probabilities underlying all oracles Ω_i are known (we discuss ways to initialize and update these probabilities in the subsequent section). Recalling that we assume that our oracles are independent, we can now approximate the probability that $l \equiv y$ (with $y \in \{\top, \bot\}$) for any given link candidate l using the following Bayesian model:

$$p(l{=}y|\overrightarrow{w}{=}\overrightarrow{x}){=}\frac{\prod_{i=1}^{n} p(\omega_i{=}x_i|l{\equiv}y)}{\prod_{i=1}^{n} p(\omega_i{=}x_i)} p(l{\equiv}y) \tag{1}$$

Recall that the *odds* of an event A occurring are defined as $odds(A) = p(A)/P(\neg A)$. For example, the odds of any link candidate being a correct link (denoted o^+) are given by

$$o^+ = \frac{p(l \equiv \top)}{p(l \equiv \bot)} \text{ for any } l \in LC. \tag{2}$$

o^+ is independent of l and stands for the odds that an element of LC chosen randomly would be a link. Given feedback from our oracles, we can approximate the odds of a link candidate l being a correct link by computing the following:

$$odds(l{\equiv}\top|\overrightarrow{w}{=}\overrightarrow{x}){=}\left(\prod_{i=1}^{n}\frac{p(\omega_i{=}x_i|l{\equiv}\top)}{p(\omega_i{=}x_i|l{\equiv}\bot)}\right)\frac{p(l{\equiv}\top)}{p(l{\equiv}\bot)}{=}\left(\prod_{i=1}^{n}\frac{p(\omega_i{=}x_i|l{\equiv}\top)}{p(\omega_i{=}x_i|l{\equiv}\bot)}\right)o^+. \tag{3}$$

A key idea behind our model is that a link candidate l can be considered to be a correct link if $odds(l \equiv \top|\overrightarrow{w} = \overrightarrow{x}) \geq k$ with $k > 1$. A link candidate is assumed to not be a link if $odds(l|\overrightarrow{w} = \overrightarrow{x}) \leq 1/k$. All other link candidates remain unclassified. Computing the odds for a link now boils down to (1) approximating the four probabilities which characterize our oracles and (2) computing o^+. As known from previous works on probabilistic models [7], o^+ is hard to compute directly as it requires knowing the set of links M, which is exactly what we are trying to compute. Several strategies can be used to approximate o^+. In this work, we consider the following three:

1. *Ignore strategy:* We can assume the probabilities $p(l = \top)$ and $p(l = \bot)$ to be equally unknown and hence use $o^+ = 1$. This reduces Equation 3 to

$$odds(l{=}\top|\overrightarrow{w}{=}\overrightarrow{x}){=}\prod_{i=1}^{n}\frac{p(\omega_i{=}x_i|l{\equiv}\top)}{p(\omega_i{=}x_i|l{\equiv}\bot)}. \tag{4}$$

Algorithm 1: LIGON Learning Algorithm

Input: Set of positive examples $E_0 \subseteq LC$; Oracles $\Omega_1 \dots \Omega_n$; Odds parameter k

$j \longleftarrow 0$;
foreach *oracle* Ω_i **do**
  Initialize confusion matrix C_i with $\frac{1}{2}$;
repeat
  foreach *oracle* Ω_i **do**
    Gather $\omega_i(l)$ for each $l \in E_j$;
    Update the confusion matrix C_i ;
    Update the characteristic matrix D_i;
  Train ACTIVE LEARNER (WOMBAT by default) using $\bigcup_{i=0}^{j} E_i$;
  Compute the set of the most informative unlabeled examples E^*;
  foreach *link candidate* $l \in E^*$ **do**
    Get the oracle result vector $\overrightarrow{x}$ for l;
  Compute the set E^+ of positive examples with $odds(l = \top | \overrightarrow{w} = \overrightarrow{x}) \geq k$;
  Compute the set E^- of negative examples with $odds(l = \top | \overrightarrow{w} = \overrightarrow{x}) \leq \frac{1}{k}$;
  $j \longleftarrow j + 1$;
  $E_j \longleftarrow E^+ \cup E^-$;
until *termination criterion holds*;
return best link specification;

2. *Equivalence strategy:* If r is an equivalence relation (e.g., `owl:sameAs`), then the set of all possible candidates has the size $|S||T|$. There can be at most $\min(|S|, |T|)$ links between S and T as no two pairs (s, t) and (s, t') can be linked if $t \neq t'$ and vice-versa (see [13]). Hence,

$$o^+ \approx \frac{\min(|S|, |T|)}{|S||T| - \min(|S|, |T|)}. \tag{5}$$

3. *Approximate strategy:* We approximate o^+ by using our learning approach. We select the mapping $[[L^*]]$ computed using the best specification L^* learned by LIGON (see the subsequent Section) as our best current approximation of the mapping we are trying to learn. o^+ is then computed as follows:

$$o^+ \approx \frac{|[[L^*]]|}{|S||T| - |[[L^*]]|}. \tag{6}$$

We quantify the effect of these strategies on our learning algorithm in our experiments.

4 The LIGON approach

LIGON is an active learning algorithm designed to learn LS from noisy oracles. An overview of the approach is given in Algorithm 1 and explained in the sections below.

Confusion Matrices. We begin by assuming that we are given an initial set $E_0 \subseteq LC$ of positive and negative examples for links. In the first step, we aim to compute the initial approximations of the conditional probabilities which describe each of the oracles Ω_i. To this end, each oracle is assigned a confusion matrix C_i of dimension 2×2 (see lines 2-3 of Algorithm 1). Each entry of the matrix is initialized with $\frac{1}{2}$ to account for potential sampling biases due to high disparities between conditional probabilities. The first and second row of each C_i contains counts for links where the oracle returned `true` resp. `false`. The first and second column of C_i contain *counts* for positive resp. negative examples. Hence, C_{11} basically contains counts for positive examples that were rightly classified as $\top$ by the oracle. In each learning iteration, we update the confusion matrix by presenting the oracle with unseen link candidates and incrementing the entries of C (see lines 4-8 of Algorithm 1). We discuss the computation of the training examples in the subsequent section. Based on the confusion matrix, we can approximate all conditional probabilities necessary to describe the oracle by computing the 2×2 matrix D with $d_{ij} = c_{ij}/(c_{1j} + c_{2j})$. For example, $d_{11} \approx p(\omega_i(l) = \texttt{true} | l \equiv \top)$. We call D the *characteristic matrix* of Ω.

Example 2. Imagine an oracle were presented with a set of 5 positive and 5 negative training examples, of which he classified 4 resp. 3 correctly. We get $\mathrm{C} = \begin{bmatrix} \frac{9}{2} & \frac{5}{2} \\ \frac{3}{2} & \frac{7}{2} \end{bmatrix}$ and $D = \begin{bmatrix} \frac{9}{12} & \frac{5}{12} \\ \frac{3}{12} & \frac{7}{12} \end{bmatrix}$.

Updating the Characteristic Matrices. Updating the probabilities is done via the confusion matrices. In each learning iteration, we present all oracles with the link candidates deemed to be most informative. Based on the answers of the oracles, we compute the odds for each of these link candidates. Link candidates l with odds in $[0, 1/k]$ and $[k, +\infty[$ are considered to be false respectively true. The new classifications are subsequently used to update the counts in the confusion matrices and therewith also the characteristic matrix of each of the oracles.

Active Learning Approach. So far, we have assumed the existence of an active learning solution for link discovery. Several active learning approaches have been developed over recent years [8]. Of these approaches, solely those based on genetic programming can generate specifications of arbitrary complexity. However, genetic programming approaches are not deterministic and are thus difficult to use in practical applications. Newer approaches based on iterative operators such as Wombat [17] have been shown to perform well in classical link discovery tasks. Therefore, we implemented a generic interface to apply Ligon to several active learning algorithms, where we used the Wombat algorithm as the default active learning algorithm for Ligon. See our last set of experiments for results of applying Ligon to other state-of-the-art active learning approaches.

Selecting the Most Informative Examples. Given an active learning algorithm, we denote the set of the m best LS generated in a given iteration i as B_i. The most informative examples are those link candidates l, which maximize the decision

entropy across the elements of B_i [12]. Formally, let $[[B_i]]$ be the union of the set of link candidates generated by all LS $b \in B_i$. Then, the most informative link candidates are the $l \in B_i$ which maximize the entropy function $e(l, B_i)$, which is defined as follows: Let $p(l, B_i)$ be the probability that a link candidate belongs to $[[b]]$ for $b \in B_i$. Then, $e(l, B_i) = -p(l, B_i) \log_2 p(l, B_i)$.

Example 3. Let us assume $|B_i| = 4$. A link candidate l returned by two of the LS in B_i would have a probability $p(l, B_i) = 0.5$. Hence, it would have an entropy $e(l, B_i) = 0.5$.

Termination Criterion. LIGON terminates after a set number of iterations has been achieved or if a link specification learned by WOMBAT achieves an F-measure of 1 on the training data.

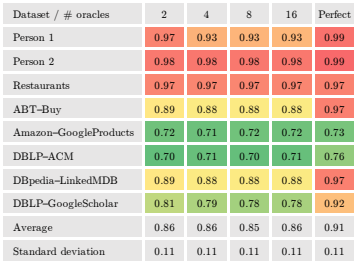
5 Experiments and Results

We aimed to answer 6 research questions with our experimental evaluation: Q_1. Which combination of strategies for computing odds and the threshold k leads to the best performance?, Q_2. How does LIGON behave when provided with an increasing number of noisy oracles?, Q_3. How well does LIGON learn from noisy oracles?, Q_4. How well does LIGON scale?, Q_5. How well does LIGON perform compare to batch learning approaches trained with a similar number of examples? and Q_6. How general is LIGON, i.e., can LIGON be applied to problems outside the link discovery domain? and does LIGON depend on the underlying active learning algorithm?

Experimantal Setup. All experiments were carried out on a 64-core 2.3 GHz PC running *OpenJDK* 64-Bit Server 1.8.0_151 on *Ubuntu* 16.04.3 LTS. Each experiment was assigned 20 GB RAM. We evaluated LIGON using 8 link discovery benchmark datasets. Five of these benchmarks were real-world datasets [6] while three were synthetic from the OAEI 2010 benchmark.[6] We used the paradigm proposed by [5] and measured the performance of algorithms using the best F-measure they achieved. As this measure fails to capture the average behaviour of algorithm over several iterations, we also report the normalized average area under the F-measure curve, which we denote AUC. We initialized LIGON with 10 positive examples (ergo, $|E_0| = 10$). We fixed the number of the most informative examples to be labeled by the noisy oracles at each iteration to 10. For labeling the most informative examples, we use $n = 2, 4, 8$ and 16 noisy oracles which were all initialized with random confusion matrices. We set the size of B to 10. All experiments were repeated 10 times and we report average values. The characteristic matrices C of our noisy oracles were generated at random. To this end we generated the true positive and true negative probabilities using a uniform distribution between 0.5 and 1, i.e. $p(\omega_i(l) = \texttt{true} | l \equiv \top) \in [0.5, 1]$ and $p(\omega_i(l) = \texttt{true} | l \equiv \top) \in [0.5, 1]$. The other probabilities were set accordingly, as they are complementary to the former two.

[6] http://oaei.ontologymatching.org/2010

Fig. 2: Average AUC heatmap of LIGON. using $2, 4, 8$ and 16 noisy oracles and the perfect oracle.



Dataset / # oracles	2	4	8	16	Perfect
Person 1	0.97	0.93	0.93	0.93	0.99
Person 2	0.98	0.98	0.98	0.98	0.99
Restaurants	0.97	0.97	0.97	0.97	0.97
ABT-Buy	0.89	0.88	0.88	0.88	0.97
Amazon-GoogleProducts	0.72	0.71	0.72	0.72	0.73
DBLP-ACM	0.70	0.71	0.70	0.71	0.76
DBpedia-LinkedMDB	0.89	0.88	0.88	0.88	0.97
DBLP-GoogleScholar	0.81	0.79	0.78	0.78	0.92
Average	0.86	0.86	0.85	0.86	0.91
Standard deviation	0.11	0.11	0.11	0.11	0.11

Table 2: Average learning iteration runtime analysis (in seconds).

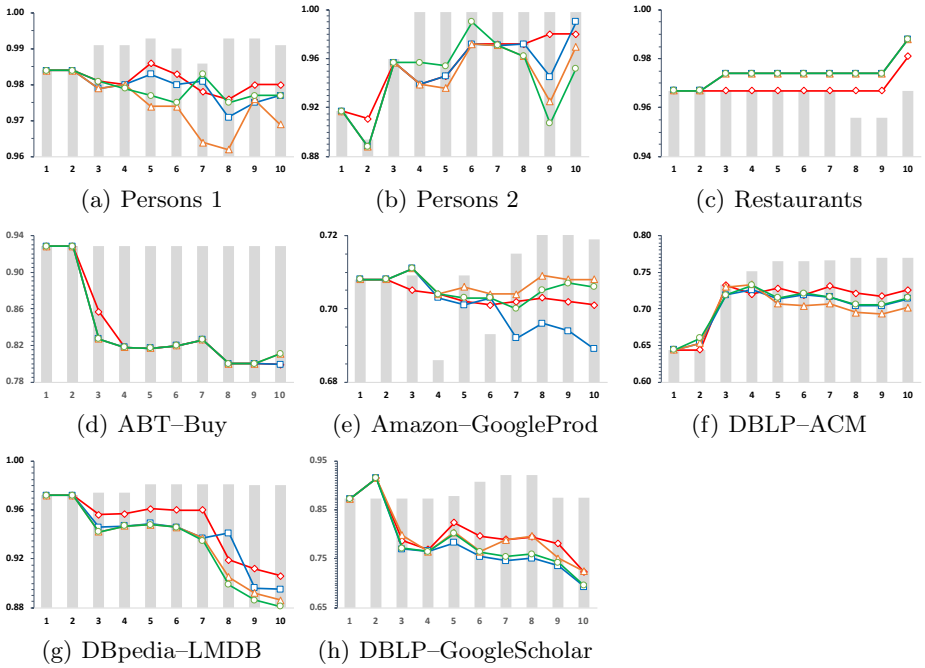
Datasets	LIGON	WOMBAT
Persons 1	2.415	2.412
Persons 2	0.946	0.942
Restaurants	0.261	0.258
ABT–Buy	4.277	4.273
Amazon–GoogleProd	2.848	2.844
DBLP–ACM	4.277	4.273
DBpedia–LinkedMDB	6.158	6.154
DBLP–GoogleScholar	16.072	16.067

Parameter Estimation. Our first series of experiments aimed to answer Q_1. We ran LIGON with $k = 2, 4, 8$ and 16. These settings were used in combination with all three strategies for computing o^+ aforementioned. A first observation is that the AUC achieved by LIGON does not depend much on the value of k nor on the strategy used. This is a highly positive feature of our algorithm as it suggests that our approach is robust w.r.t. to how it is initialized. Interestingly, this experiment already suggests that LIGON achieves more than 95% of the performance of the original WOMBAT algorithm trained with a perfect oracle.

We chose to run the remainder of our experiments with the setting $k = 16$ combined with the *equivalent* strategy as this combination achieved the highest average F-measure of 0.86.

Comparison with Perfect Oracle. In our second set of experiments, we answered Q_2 by measuring how well LIGON performed when provided with an increasing number of oracles. In this series of experiments, we used 2, 4, 8, and 16 oracles which were initialized randomly. k was set to 16 and we used the *Equivalent* strategy. Once more, the robustness of our approach became evident as its performance was not majorly perturbed by a variation in the number of oracles. In all settings LIGON achieves an average AUC close to 0.86 with no statistical difference. We can hence conclude that the performance of our approach depends mostly on the initial set of examples E_0 being accurate, which leads to our prior—i.e., the evaluation of the initial confusion matrix of the oracles—being sufficient. This sufficient approximation means that our Bayesian model is able to distinguish between erroneous classifications well enough to find most informative examples accurately and generalize over them. In other words, even a small balanced set containing 5 positive and 5 negative examples seems sufficient to approximate the confusion matrix of the oracles sufficiently well to detect positive and negative examples consistently in the subsequent steps. This answers Q_2. Figures 2 and 3 show the detailed results of running LIGON for 10 iterations for each of our 8 benchmark datasets.

To answer Q_3, we also ran our approach in combination with a *perfect oracle* (i.e., an oracle which knew and returned the perfect classification for all pairs



(a) Persons 1 (b) Persons 2 (c) Restaurants

(d) ABT–Buy (e) Amazon–GoogleProd (f) DBLP–ACM

(g) DBpedia–LMDB (h) DBLP–GoogleScholar

Fig. 3: F-measure results of LIGON. x-axes show the iteration number while the y-axes show the F-measure. Note that, the y-axes show different value for better legibility. Gray bars represent the F-measure of LIGON with the perfect oracle while the F-measure achieved by the 2, 4, 8 and 16 noisy oracles are represented by red, blue, orange and green lines respectively.

from (S, T)). The detailed results are provided in Figures 3 and 2. Combining our approach with a perfect oracle can be regarded as providing an upper bound to our learning algorithm. Over all datasets, LIGON achieved 95% of the AUC achieved with the perfect oracle (min = 88% on *DBpedia-LMDB*, max = 100% on *Restaurants*) of the AUC achieved with the perfect oracle. This answers Q_3 and demonstrates that LIGON can learn LS with an accuracy close to that of an approach provided with perfect answers.

Runtime. In our third set of experiments, we were interested in knowing how well our approach scales. To this end, we measured the runtime of our algorithm while running the experiments carried out to answer Q_2 and Q_3. In our experiments, WOMBAT, the machine learning approach used within LIGON, makes up for more than 99% of LIGON's runtime. for more detailed results see Table 2. This shows that the Bayesian framework used to re-evaluate the characteristic matrices of the oracles is clearly fast enough to be used in interactive scenarios, which answers Q_4. Our approach completes a learning iteration in less than 10 seconds on most datasets, which we consider acceptable even for interac-

Dataset	Pessimistic	Reweighted	Simple	Complete	LIGON
DBLP-ACM	0.93	**0.95**	0.94	0.94	0.73
Amazon-GoogleProduct	0.39	0.43	0.53	0.45	**0.71**
ABT-Buy	0.36	0.37	0.37	0.36	**0.93**
Average	0.77	0.78	0.77	0.74	**0.89**

Table 3: F-Measure achieved by LIGON vs. State of the art from [5] and [17].

tive scenarios. The longer runtime on *DBLP-GoogleScholar* (roughly 16 seconds per iteration on average) is due to the large size of this dataset. Here, a parallel version of the WOMBAT algorithm would help improving the interaction with the user. The implementation of a parallel version of WOMBAT goes beyond the work presented here.

Comparison with Batch Learning. While active learning commonly requires a small number of training examples to achieve good F-measures, other techniques such as pessimistic and re-weighted batch learning have also been designed to achieve this goal [5]. In addition, the positive-only learning algorithm WOMBAT has also been shown to perform well with a small number of training examples. In our final set of experiments, we compared the best F-measure achieved by LIGON when trained with 16 noisy oracles, $k = 16$ and the *equivalent* strategy with the pessimistic and re-weighted models proposed in [5] as well as the two versions of the WOMBAT approach [17]. All approaches were trained with 2% of the reference data (i.e., with a perfect oracle) as suggested by [5]. The results of these experiments are shown in Table 3. Note that, we did not consider the datasets *Persons 1*, *Persons 2* and *Restaurant* because 2% of the training data accounts to less than 10 examples, which LIGON requires as initial training dataset E_0. Our results answer Q_5 clearly by showing that LIGON outperforms previous batch learning algorithms even when trained with noisy oracles. On average, LIGON is more than 40% better in F-measure. This clearly demonstrates that our active learning strategy for selecting training examples is superior to batch learning.

Generalization of LIGON. In our last set of experiment, we implemented a *generalization of* LIGON *for binary classification tasks behind link discovery*. We thus used the active learning framework JCLAL [15] to wrap WEKA [2] classifiers and implemented LIGON as a custom oracle. We selected three well known binary classification datasets (i.e., *Diabetes, breast-cancer* and *Ionosphere*) from the WEKA distribution on which we applied two state-of-the-art classification algorithms, namely GBDT and Random Forests [20]. Based on our previous experiments, we used 4 noisy oracles, k was set to 16 and we used the *Ignore* strategy, since all the other strategies are specific to the link discovery domain. We executed two sets of experiments for noisy oracles with true positive/negative probabilities drawn from the two uniform distributions in $[0.5, 1]$ and $[0.75, 1]$. On average, LIGON achieves 75% and 89% of the learning accuracy for noisy oracles drawn from $[0.5, 1]$ and $[0.75, 1]$ respectively. These results indicate that

LIGON is not only applicable to problems outside the link discovery domain but also independent from the underlying active learning algorithm is able to achieve F-measures near to the ones scored using a perfect oracle, which answers Q_6.

6 Related Work

Link Discovery for RDF knowledge graphs has been an active research area for nearly a decade, with the first frameworks for link discovery [4, 9] appearing at the beginning of the decade. RAVEN [10] was the first active learning approach for link discovery and used perception learning to detect accurate LS. Other approaches were subsequently developed to learn LS within the active learning setting [3, 11, 12]. Unsupervised learning approaches for monogamous relations [11–13] rely on different pseudo-F-measures to detect links without any training data. Positive-only learning algorithms [17] address the open-world characteristic of the Semantic Web by using generalization algorithms to detect LS. The work presented by [14] proposes an active learning approach for link prediction in knowledge graphs. LIGON differs from the state of the art in that it does not assume that it deals with perfect oracles. Rather, it uses several noisy oracles to achieve an F-measures close to those achieved with perfect oracles. An *active learning approach with uncertain labeling knowledge* is proposed by [1], where the authors used diversity density to characterize the uncertainty of the knowledge. A probabilistic model of active learning with multiple noisy oracles was introduced by [18] to label the data based on the most perfect oracle. For *Crowdsourcing* scenarios, [16] propose a supervised learning algorithm for multiple annotators (oracles), where the oracles' diverse reliabilities were treated as a latent variables.

7 Conclusions and Future Work

We presented LIGON, an active learning approach designed to deal with noisy oracles, i.e., oracles that are not guaranteed to return correct classification results. LIGON relies on a probabilistic model to estimate the joint odds of link candidates based on the oracles' guesses. Our experiments showed that LIGON achieves 95% of the learning accuracy of approaches learning with perfect oracles in the link discovery setting. Moreover, we showed that LIGON is (1) not dependent on the underlying active learning algorithm and (2) able to deal with other classification problems. In future work, we will evaluate LIGON within real crowdsourcing scenarios. A limitation of our approach is that it assumes that the confusion matrix of the oracles is static. While this assumption is valid with the small number of iterations necessary for our approach to converge, we will extend our model so as to deal with oracles which change dynamically. Furthermore, we will extend LIGON to handle n-ary classification problems and evaluate it on more stat-of-the-art approaches from the deep learning domain.

Acknowledgments. *This work has been supported by the EU H2020 project KnowGraphs (GA no. 860801) as well as the BMVI projects LIMBO (GA no. 19F2029C) and OPAL (GA no. 19F2028A).*

References

1. M. Fang and X. Zhu. Active learning with uncertain labeling knowledge. *Pattern Recognition Letters*, 43(Supplement C):98 – 108, 2014. ICPR2012 Awarded Papers.
2. M. A. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 2009.
3. R. Isele and C. Bizer. Active learning of expressive linkage rules using genetic programming. *J. Web Sem.*, 23:2–15, 2013.
4. A. Jentzsch, R. Isele, and C. Bizer. Silk - generating RDF links while publishing or consuming linked data. In *Proceedings of the ISWC Posters & Demons*, 2010.
5. M. Kejriwal and D. P. Miranker. Semi-supervised instance matching using boosted classifiers. In *The Semantic Web. Latest Advances and New Domains*. 2015.
6. H. Köpcke, A. Thor, and E. Rahm. Evaluation of entity resolution approaches on real-world match problems. *Proc. VLDB Endow.*, 3(1-2):484–493, Sept. 2010.
7. C. D. Manning, P. Raghavan, and H. Schütze. Introduction to information retrieval, 2008.
8. M. Nentwig, M. Hartung, A. N. Ngomo, and E. Rahm. A survey of current link discovery frameworks. *Semantic Web*, 8(3):419–436, 2017.
9. A. N. Ngomo and S. Auer. LIMES - A time-efficient approach for large-scale link discovery on the web of data. In *Proceedings of the International Joint Conference on Artificial Intelligence, Spain, 2011*.
10. A.-C. Ngonga Ngomo, J. Lehmann, S. Auer, and K. Höffner. RAVEN - active learning of link specifications. In *Proceedings of the 6th International Workshop on Ontology Matching, Germany, 2011*.
11. A.-C. Ngonga Ngomo and K. Lyko. Eagle: Efficient active learning of link specifications using genetic programming. In *Extended Semantic Web Conference*, pages 149–163. Springer, 2012.
12. A.-C. Ngonga Ngomo, K. Lyko, and V. Christen. Coala–correlation-aware active learning of link specifications. In *Extended Semantic Web Conference*, 2013.
13. A. Nikolov, M. d'Aquin, and E. Motta. Unsupervised learning of link discovery configuration. In *Extended Semantic Web Conference*. Springer, 2012.
14. N. Ostapuk, J. Yang, and P. Cudré-Mauroux. Activelink: deep active learning for link prediction in knowledge graphs. In *The World Wide Web Conference*, 2019.
15. O. G. R. Pupo, E. Pérez, M. del Carmen Rodríguez-Hernández, H. M. Fardoun, and S. Ventura. JCLAL: A java framework for active learning. *J. Mach. Learn. Res.*, 2016.
16. F. Rodrigues, F. Pereira, and B. Ribeiro. Learning from multiple annotators: Distinguishing good from random labelers. *Pattern Recognition Letters*, 2013.
17. M. Sherif, A.-C. Ngonga Ngomo, and J. Lehmann. WOMBAT - A Generalization Approach for Automatic Link Discovery. In *14th Extended Semantic Web Conference, Slovenia*. Springer, 2017.
18. W. Wu, Y. Liu, M. Guo, C. Wang, and X. Liu. A probabilistic model of active learning with multiple noisy oracles. *Neurocomputing*, 2013.
19. H. Yu, Z. Shen, C. Miao, and B. An. Challenges and opportunities for trust management in crowdsourcing. In *Proceedings of the The 2012 IEEE/WIC/ACM WI-IAT '12*, USA, 2012.
20. C. Zhang, C. Liu, X. Zhang, and G. Almpanidis. An up-to-date comparison of state-of-the-art classification algorithms. *Expert Systems with Applications*, 2017.

Supervised Ontology and Instance Matching with MELT

Sven Hertling[1⋆[0000−0003−0333−5888]], Jan Portisch[1,2⋆[0000−0001−5420−0663]], and Heiko Paulheim[1[0000−0003−4386−8195]]

[1] Data and Web Science Group, University of Mannheim, Germany
{jan, sven, heiko}@informatik.uni-mannheim.de
[2] SAP SE Product Engineering Financial Services, Walldorf, Germany
{jan.portisch}@sap.com

Abstract. In this paper, we present *MELT-ML*, a machine learning extension to the *Matching and EvaLuation Toolkit* (MELT) which facilitates the application of supervised learning for ontology and instance matching. Our contributions are twofold: We present an open source machine learning extension to the matching toolkit as well as two supervised learning use cases demonstrating the capabilities of the new extension.

Keywords: ontology matching · supervised learning · machine learning · knowledge graph embeddings

1 Introduction

Many similarity metrics and matching approaches have been proposed and developed up to date. They are typically implemented as engineered systems which apply a process-oriented matching pipeline. Manually combining metrics, also called *features* in the machine learning jargon, is typically very cumbersome. Supervised learning allows researchers and developers to focus on adding and defining features and to leave the weighting of those and the decision making to a machine. This approach may also be suitable for developing generic matching systems that self-adapt depending on specific datasets or domains. Here, it makes sense to test and evaluate multiple classifiers at once in a fair, i.e. reproducible, way. Furthermore, recent advances in machine learning – such as in the area of knowledge graph embeddings – may also be applicable for the ontology and instance matching community. The existing evaluation and development platforms, such as the *Alignment API* [3], *SEALS* [7,33] or the *HOBBIT* [25] framework, make the application of such advances not as simple as it could be.

⋆ The authors contributed equally to this paper.

In this paper, we present *MELT-ML*, an extension to the *Matching and EvaLuation Toolkit* (MELT). Our contribution is twofold: Firstly, we present a machine learning extension to the MELT framework (available in MELT 2.6) which simplifies the application of advanced machine learning algorithms in matching systems and which helps researchers to evaluate systems that exploit such techniques. Secondly, we present and evaluate two novel approaches in an exemplary manner implemented and evaluated with the extension in order to demonstrate its functionality. We show that RDF2Vec [30] embeddings derived directly from the ontologies to be matched are capable of representing the internal structure of an ontology but do not provide any value for matching tasks with differently structured ontologies when evaluated as the only feature. We further show that multiple feature generators and a machine learning component help to obtain a high precision alignment in the *Ontology Alignment Evaluation Initiative* (OAEI) *knowledge graph* track [11,8].

2 Related Work

Classification is a flavor of *supervised learning* and denotes a machine learning approach where the learning system is presented with a set of records carrying a *class* or *label*. Given those records, the system is trained by trying to predict the correct class. [18] Transferred to the ontology alignment domain, the set of records can be regarded as a collection of correspondences where some of the correspondences are correct (class *true*) and some correspondences are false (class *false*). Hence, the classification system at hand is binary.

The application of supervised learning is not new to ontology matching. In fact, even in the very first edition of the OAEI[3] in 2004 the *OLA* matching system [5] performed a simple optimization of weights using the provided reference alignments. In the past, multiple publications [14,4,31,24,16] addressed supervised learning in ontology matching, occasionally also referred to as *matching learning*. Unsupervised machine learning approaches are less often used, but have been proposed for the task of combining matchers as well [23].

More recently, Nkisi-Orji et al. [26] present a matching system that uses a multitude of features and a random forest classifier. The system is evaluated on the OAEI *conference* track [2] and the EuroVoc dataset, but did not participate in the actual evaluation campaign. Similarly, Wang et al. [32] present a system called *OntoEmma* which exploits a neural classifier together with 32 features. The system is evaluated on the *large biomed* track. However, the system did not participate in an OAEI campaign either. It should be mentioned here that a comparison between systems that have been trained with parts of the reference and systems that have not is not really fair (despite being the typical approach).

Also a recent, OAEI-participating matching system applies supervised learning: The *POMap++* matching system [16] uses a local classifier which is not

[3] Back then the competition was actually referred to as *EON Ontology Alignment Contest*.

based on the reference alignment but on a locally created gold standard. The system also participated in the last two recent OAEI campaigns [17,15].

The implementations of the approaches are typically not easily reusable or available in a central framework.

3 The MELT Framework

Overview MELT [10] is a framework written in Java for ontology and instance matcher development, tuning, evaluation, and packaging. It supports both, HOBBIT and SEALS, two heavily used evaluation platforms in the ontology matching community. The core parts of the framework are implemented in Java, but evaluation and packaging of matchers implemented in other languages is also supported. Since 2020, MELT is the official framework recommendation by the OAEI and the MELT track repository is used to provide all track data required by SEALS. MELT is also capable of rendering Web dashboards for ontology matching results so that interested parties can analyze and compare matching results on the level of correspondences without any coding efforts [27]. This has been pioneered at the OAEI 2019 for the *knowledge graph* track.[4] MELT is open-source[5], under a permissive license, and is available on the maven central repository[6].

Different Gold Standard Types Matching systems are typically evaluated against a reference alignment. A reference alignment may be complete or only partially complete. The latter means that not all entities in the matching task are aligned and that any entity not appearing in the gold standard cannot be judged. Therefore, the following five levels of completeness can be distinguished: (i) complete, (ii) partial with complete target and complete source, (iii) partial with complete target and incomplete source, (iv) partial with complete source and incomplete target, (v) partial with incomplete source and incomplete target. If the reference is complete, all correspondences not available in the reference alignment can be regarded as wrong. If only one part of the gold standard is complete (ii, iii, and iv), every correspondence involving an element of the complete side that is not available in the reference can be regarded as wrong. If the gold standard is incomplete (v), the correctness of correspondences not in the gold standard cannot be judged. For example, given that the gold standard is partial with complete target and complete source (case ii), and given the correspondence $< a, b, =, 1.0 >$, the correspondence $< a, c, =, 1.0 >$ could be judged as wrong because it involves a which is from the complete side of the alignment. On the other hand, the correspondence $< d, e, =, 1.0 >$ cannot be judged because it does not involve any element from the gold standard. This evaluation setting is used for example for the OAEI *knowledge graph* track. OAEI reference datasets are typically complete

[4] For a demo of the MELT dashboard, see `https://dwslab.github.io/melt/anatomy_conference_dashboard.html`

[5] `https://github.com/dwslab/melt/`

[6] `https://mvnrepository.com/artifact/de.uni-mannheim.informatik.dws.melt`

with the exception of the *knowledge graph* track. The completeness of references influences how matching systems have to be evaluated. MELT can handle all stated levels of completeness. The completeness can be set for every `TestCase` separately using the enum `GoldStandardCompleteness`. The completeness also influences the generation of negative correspondences for a gold standard in supervised learning. MELT supports matching system developers also in this use case.

4 Supervised Learning Extensions in MELT

4.1 Python Wrapper

As researchers apply advances in machine learning and natural language processing to other domains, they often turn to Python because leading machine learning libraries such as *scikit-learn*[7], *TensorFlow*[8], *PyTorch*[9], *Keras*[10], or *gensim*[11] are not easily available for the Java language. In order to exploit functionalities provided by Python libraries in a consistent manner without a tool break, a wrapper is implemented in MELT which communicates with a Python backend via HTTP as depicted in Figure 1. The server works out-of-the-box requiring only that Python and the libraries listed in the `requirements.txt` file are available on the target system. The MELT-ML user can call methods in Java which are mapped to a Python call in the background. As of MELT 2.6, functionality from *gensim* and *scikit-learn* are wrapped.

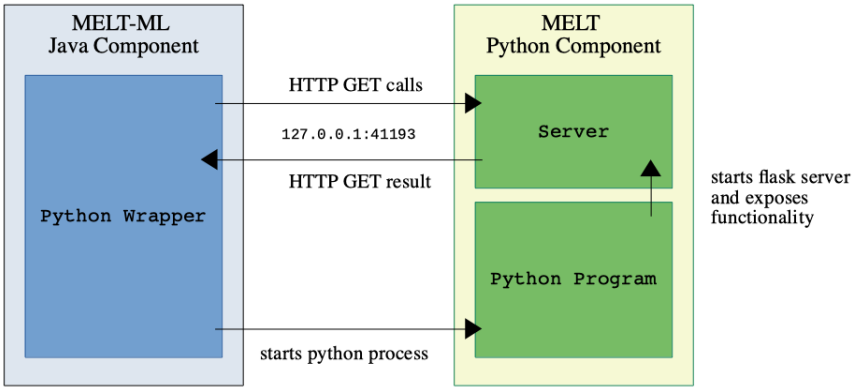


Fig. 1. Python code execution in MELT.

[7] `https://scikit-learn.org/`
[8] `https://www.tensorflow.org/`
[9] `https://pytorch.org/`
[10] `https://keras.io/`
[11] `https://radimrehurek.com/gensim/`

4.2 Generation of Training Data

Every classification approach needs features and class labels. In the case of matching, each example represents a correspondence and the overall goal is to have an ML model which is capable of deciding if a correspondence is correct or not. Thus, the matching component can only work as a filter e.g. it can only remove correspondences of an already generated alignment.

For training such a classifier, positive and negative examples are required. The positive ones can be generated by a high precision matcher or by an externally provided alignment such as a sample of the reference alignment or manually created correspondences. As mentioned earlier, no OAEI track provides a dedicated alignment for training. Therefore, MELT provides a new `sample(int n)` method in the `Alignment` class for sampling n correct correspondences as well as `sampleByFraction(double fraction)` for sampling a $fraction$ in range $(0, 1)$ of correct correspondences.

Negative examples can be easily generated in settings where the gold standard is complete or partially complete (with complete source and/or target, see Section 3). The reason is that any correspondence with an entity appearing in the positive examples can be regarded as incorrect. Thus, a recall oriented matcher can generate an alignment and all such correspondences represent the negative class. In cases where the gold standard is partial and the source and/or target is incomplete, each negative correspondence has to be manually created.

4.3 Generation of Features

The features for the correspondences are generated by one or more matchers which can be concatenated in a pipeline or any other control flow. MELT provides an explicit framework for storing the feature values in correspondence extensions (which are by default also serialized in the alignment format). The correspondence method `addAdditionalConfidence(String key, double confidence)` is used to add such feature values (more convenience methods exist).

MELT already provides some out-of-the-box feature generators in the form of so called *filters* and *matchers*. A *matcher* detects new correspondences. As of MELT 2.6, 17 matchers are directly available (e.g., different string similarity metrics). A *filter* requires an input alignment and adds the additional confidences to the correspondences, or removes correspondences below a threshold. In MELT, machine learning is also included via a filter (`MachineLearningScikitFilter`). As of MELT 2.6, 21 filters are available. A selection is presented in the following:

SimilarNeighboursFilter Given an initial alignment of instances, the `SimilarNeighboursFilter` analyzes for each of the instance correspondences how many already matched neighbours the source and target instances share. It can be further customized to also include similar literals (defined by string processing methods). The share of neighbours can be added to the correspondence as absolute value or relative to the total numbers of neighbours for source and target. For the latter, the user can choose from `min` (size of the intersection divided by

minimum number of neighbours of source or target), `max`, `jaccard` (size of intersection dived by size of union), and `dice` (twice the size of intersection divided by the sum of source and target neighbours).

CommonPropertiesFilter This filter selects instance matches based on the overlap of properties. The idea is that equal instances also share similar properties. Especially in the case of homonyms, this filter might help. For instance, given two instances with label 'bat', the string may refer to the mammal or to the racket where the first sense has properties like 'taxon', 'age', or 'habitat' and the latter one has properties like 'material', 'quality', or 'producer'. This filter of course requires already matched properties. The added confidence can be further customized similarly to the previous filter. Furthermore, property URIs are by default filtered to exclude properties like `rdfs:label`.

SimilarHierarchyFilter This component analyzes any hierarchy for given instance matches such as type hierarchy or a category taxonomy as given in the *knowledge graph* track. Thus, two properties are needed: 1) instance to hierarchy property which connects the instance to the hierarchy (in case of type hierarchy this is `rdf:type`) 2) hierarchy property which connects the hierarchy (in case of type hierarchy this is `rdfs:subClassOf`). This filter needs matches in the hierarchy which are counted similarly to the previous filters. Additionally, the confidence can be computed by a hierarchy level dependent value (the higher the match in the hierarchy, the lower the confidence). `SimilarTypeFilter` is a reduced version of it by just looking at the direct parent.

BagOfWordsSetSimilarityFilter This filter analyzes the token overlap of the literals given by a specific property. The tokenizer can be freely chosen as well as the overlap similarity.

MachineLearningScikitFilter The actual classification part is implemented in class `MachineLearningScikitFilter`. In the standard setting, a five-fold cross validation is executed to search for the model with the best f-measure. The following models and hyper parameters are tested:

- *Decision Trees* optimized by minimum leaf size and maximum depth of tree (1-20)
- *Gradient Boosted Trees* optimized by maximum depth (1,6,11,16,21) and number of trees (1,21,41,61,81,101)
- *Random Forest* optimized by number of trees (1-100 with 10 steps) and minimum leaf size (1-10)
- *Naïve Bayes* (without specific parameter tuning)
- *Support Vector Machines* (SVM) with radial base function kernel; C and gamma are tuned according to [13]
- *Neural Network* with one hidden layer in two different sizes $F/2+2$, $sqrt(F)$, and two hidden layers of $F/2$ and $sqrt(F)$, where F denotes the number of features

All of these combinations are evaluated automatically with and without feature normalization (`MinMaxScaler` which scales each feature to a range between zero and one). The best model is then trained on the whole training set and applied to the given alignment.

4.4 Analysis of Matches

A correspondence which was found by a matching system and which appears in the reference alignment is referred to as *true positive*. A *residual true positive* correspondence is a true positive correspondence that is not trivial as defined by a trivial alignment. The trivial alignment can be given or calculated by a simple baseline matcher. String matches, for instance, are often referred to as trivial. Given a reference alignment, a system alignment, and a trivial alignment, the *residual recall* can be calculated as the share of non trivial correspondences found by the matching system [1,6].

If a matcher was trained using a sample of the reference alignment and is also evaluated on the reference alignment, a true positive match can only be counted as meaningful if it was not available in the training set before. In MELT, the baseline matcher can be set dynamically for an evaluation. Therefore, for supervised matching tasks where a sample from the reference is used, the sample can be set as baseline solution (using the `ForwardMatcher`) so that only additionally found matches are counted as residual true positives. Using the alignment cube file[12], residual true positives can be analyzed at the level of individual correspondences.

5 Exemplary Analysis

5.1 RDF2Vec Vector Projections

Experiment In this experiment, the ontologies to be matched are embedded and a projection is used to determine matches. *RDF2Vec* is a knowledge graph embedding approach which generates random walks for each node in the graph to be embedded and afterwards runs the *word2vec* [21,22] algorithm on the generated walks. Thereby, a vector for each node in the graph is obtained. The RDF graph is used in *RDF2Vec* without any pre-processing such as in other approaches like *OWL2Vec* [12]. The embedding approach chosen here has been used on external background knowledge for ontology alignment before [29].

In this setting, we train embeddings for the ontologies to be matched. In order to do so, we integrate the *jRDF2Vec*[13] [28] framework into MELT in order to train the embedding spaces. Using the functionalities provided in the MELT-ML package, we train a linear projection from the source vector space into the target vector space. In order to generate a training dataset for the projection,

[12] The alignment cube file is a CSV file listing all correspondences found and not found (together with filtering properties) that is generated by the `EvaluatorCSV`.

[13] `https://github.com/dwslab/jRDF2Vec`

the `sampleByFraction(double fraction)` method is used. For each source, the closest target node in the embedding space is determined. If the confidence for a match is above a threshold t, the correspondence is added to the system alignment.

Here, we do not apply any additional matching techniques such as string matching. The approach is fully independent of any stated label information. The exemplary matching system is available online as an example.[14]

Results For the vector training, we generate 100 random walks with a depth of 4 per node and train skip-gram (SG) embeddings with 50 dimensions, minimum count of 1, and a window size of 5. We use a sampling rate of 50% and a threshold of 0.85. While the implemented matcher fails to generate a meaningful residual recall when the two ontologies to be matched are different, it performs very well when the ontologies are of the same structure as in the *multifarm* track. Here, the approach generates many residual true positives with a residual recall of up to 61% on *iasted-iasted* as seen in Table 1. Thus, it could be shown that *RDF2Vec* embeddings do contain structural information of the knowledge graph that is embedded.

Multifarm Test Case	P	R	R+	F	# of TP	# of FP	# of FN
iasted-iasted	0.8232	0.7459	0.6111	0.7836	135	29	46
conference-conference	0.7065	0.5285	0.1967	0.6047	65	27	58
confOf-confOf	0.9111	0.5541	0.1081	0.6891	41	4	33

Table 1. Performance of RDF2Vec projections on the same ontologies in the multifarm track. *P* stands for *precision*, *r* stands for *recall*, and *R+* for *residual recall*. R+ refers here to the fraction of correspondences found that were previously not available in the training set. *# of ...* refers to the number of *true positives (TP)*, *false positives (FP)*, and *false negatives (FN)*. Details about the track can be found in [19]

5.2 Knowledge Graph Track Experiments

Experiment In this experiment, the instances of the OAEI *knowledge graph* track are matched. First, a basic matcher (`BaseMatcher`) is used to generate a recall oriented alignment by applying simple string matching on the property values of `rdfs:label` and `skos:altLabel`. The text is compared once using string equality and once in a normalized fashion (non-ASCII characters are removed and the whole string is lowercased).

Given this alignment, the above described feature generators / filters are applied in isolation to re-rank the correspondences and afterwards the `NaiveDescendingExtractor` [20] is used to create a one-to-one alignment based on the best confidence.

In contrast to this, another supervised approach is tried out. After executing the `BaseMatcher`, all feature generators are applied after each other where each

[14] `https://github.com/dwslab/melt/tree/master/examples/RDF2VecMatcher`

filter adds one feature value. The feature values are calculated independently of each other. This results in an alignment where each correspondence has the additional confidences in its extensions. As a last step, the `MachineLearningScikitFilter` is executed. The training alignment is generated by sampling all correspondences from the `BaseMatcher` where the source *or* target is involved. The correspondence is a positive training example if the source *and* the target appear in the input alignment (which is in our case the sampled reference alignment) and a negative example in all other cases.

The search for the machine learning model is executed as a five-fold cross validation and the best model is used to classify all correspondences given by the `BaseMatcher`. The whole setup is available on GitHub[15].

Results In all filters, the absolute number of overlapping entities are used (they are normalized during a grid search for the best model). In the `SimilarNeighboursFilter`, the literals are compared with text equality and the hierarchy filter compares the categories of the Wiki pages. The `SimilarTypeFilter` analyzes the direct classes which are extracted from templates (indicated by the text 'infobox'). The results for this experiment are depicted in Table 2 which shows that not one feature can be used for all test cases because different Wiki combinations (test cases) require different filters. The `BaseMatcher` already achieves a good f-measure which is also in line with previous analyses [9]. When executing the `MachineLearningScikitFilter` the precision can be increased for three test cases and the associated drop in recall is relatively small. It can be further seen that there is not one single optimal classifier out of the classifiers tested.

6 Conclusion and Outlook

With MELT-ML, we have presented a machine learning extension for the MELT framework which facilitates feature generation and feature combination. The latter are included as *filters* to refine existing matches. MELT also allows for the evaluation of ML-based matching systems.

In the future, we plan to extend the provided functionality by the Python wrapper to further facilitate machine learning in matching applications. We further plan to extend the number of feature generators. With our contribution we hope to encourage OAEI participants to apply and evaluate supervised matching techniques. In addition, we intend to further study different strategies and ratios for the generation of negative examples.

We further would like to emphasize that a special machine learning track with dedicated training and testing alignments might benefit the community, would increase the transparency in terms of matching system performance, and might further increase the number of participants since researchers use OAEI datasets for supervised learning but there is no official channel to participate if parts of the reference alignment are required.

[15] `https://github.com/dwslab/melt/tree/master/examples/supervisedKGTrackMatcher`

	mcu-marvel			memoryalpha-memorybeta			memoryalpha-stexpanded			starwars-swg			starwars-swtor		
Approach	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**	**P**	**R**	**F**
BaseMatcher	0.8548	0.6796	0.7572	0.8740	0.8978	0.8858	0.8675	0.9264	0.8960	0.9001	0.7318	0.8072	0.9007	0.9146	0.9076
CommonPropertiesFilter	0.8823	0.6614	0.7560	0.9310	0.8785	0.9040	0.9370	0.8968	0.9165	0.9257	0.7162	0.8076	0.9371	0.8999	0.9181
SimilarHierarchyFilter	0.8823	0.6614	0.7560	0.9361	0.8830	0.9088	0.9527	0.9107	0.9312	0.9281	0.7181	0.8097	0.9440	0.9057	0.9245
BagOfWordsSetSimilarityFilter	0.8823	0.6614	0.7560	0.9340	0.8810	0.9067	0.9406	0.8991	0.9194	0.9292	0.7190	0.8107	0.9348	0.8976	0.9159
SimilarNeighboursFilter	0.8912	0.6687	0.7641	0.9467	0.8916	0.9183	0.9600	0.9171	0.9380	0.9375	0.7254	0.8179	0.9317	0.8947	0.9128
SimilarTypeFilter	0.8823	0.6614	0.7560	0.9247	0.8727	0.8980	0.9303	0.8899	0.9096	0.9222	0.7135	0.8045	0.9326	0.8962	0.9140
ML (sample=0.2)	0.8831	0.6620	0.7567	0.9636	0.8592	0.9084	0.9648	0.8887	0.9252	0.9292	0.7190	0.8107	0.9621	0.8778	0.9180
	SVM			Random Forest			SVM			SVM			Random Forest		
ML (sample=0.4)	0.8831	0.6620	0.7567	0.9636	0.8599	0.9088	0.9734	0.8690	0.9182	0.9315	0.7199	0.8121	0.9445	0.8903	0.9166
	Random Forest			Random Forest			Neural Network			Neural Network			Random Forest		
ML (sample=0.6)	0.8831	0.6620	0.7567	0.9685	0.8575	0.9096	0.9667	0.8916	0.9276	0.9367	0.7153	0.8112	0.9565	0.8903	0.9222
	Random Forest			Decision Tree			Neural Network			SVM			SVM		

Table 2. Precision (P), recall (R), and f-measure (F) for all five test cases of the *knowledge graph* track using different matching approaches. Details about the track can be found in [9]. For the ML approaches, the optimal classifier (given the evaluated ones outlined in Subsection 4.3) is stated below the scores.

References

1. José Luis Aguirre, Bernardo Cuenca Grau, Kai Eckert, Jérôme Euzenat, Alfio Ferrara, Robert Willem Van Hague, Laura Hollink, Ernesto Jiménez-Ruiz, Christian Meilicke, Andriy Nikolov, et al. Results of the ontology alignment evaluation initiative 2012. 2012.
2. Michelle Cheatham and Pascal Hitzler. Conference v2.0: An uncertain version of the OAEI conference benchmark. In *ISWC 2014. Proceedings, Part II*, pages 33–48, 2014.
3. Jérôme David, Jérôme Euzenat, François Scharffe, and Cássia Trojahn dos Santos. The alignment API 4.0. *Semantic Web*, 2(1):3–10, 2011.
4. Kai Eckert, Christian Meilicke, and Heiner Stuckenschmidt. Improving ontology matching using meta-level learning. In *ESWC 2009, Proceedings*, pages 158–172, 2009.
5. Jérôme Euzenat, David Loup, Mohamed Touzani, and Petko Valtchev. Ontology alignment with OLA. In *EON 2004, Evaluation of Ontology-based Tools, Proceedings of the 3rd International Workshop on Evaluation of Ontology-based Tools held at ISWC 2004*, 2004.
6. Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching*, chapter 9, pages 285–317. Springer, New York, 2nd edition, 2013.
7. Raúl García-Castro, Miguel Esteban-Gutiérrez, and Asunción Gómez-Pérez. Towards an infrastructure for the evaluation of semantic technologies. In *eChallenges e-2010 Conference*, pages 1–7. IEEE, 2010.
8. Sven Hertling and Heiko Paulheim. Dbkwik: A consolidated knowledge graph from thousands of wikis. In *2018 IEEE International Conference on Big Knowledge, ICBK 2018, Singapore, November 17-18, 2018*, pages 17–24, 2018.
9. Sven Hertling and Heiko Paulheim. The knowledge graph track at OAEI - gold standards, baselines, and the golden hammer bias. In *ESWC 2020, Proceedings*, pages 343–359, 2020.
10. Sven Hertling, Jan Portisch, and Heiko Paulheim. MELT - matching evaluation toolkit. In *Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, Proceedings*, pages 231–245, 2019.
11. Alexandra Hofmann, Samresh Perchani, Jan Portisch, Sven Hertling, and Heiko Paulheim. Dbkwik: Towards knowledge graph creation from thousands of wikis. In *Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks, Vienna, Austria, October 23rd - to - 25th, 2017*, 2017.
12. Ole Magnus Holter, Erik Bryhn Myklebust, Jiaoyan Chen, and Ernesto Jiménez-Ruiz. Embedding OWL ontologies with owl2vec. In *ISWC 2019 Satellite Tracks (Posters & Demonstrations, Industry, and Outrageous Ideas)*, pages 33–36, 2019.
13. Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. A practical guide to support vector classification. 2003.
14. Ryutaro Ichise. Machine learning approach for ontology mapping using multiple concept similarity measures. In *Seventh IEEE/ACIS International Conference on Computer and Information Science (icis 2008)*, pages 340–346. IEEE, 2008.
15. Amir Laadhar, Faiza Ghozzi, Imen Megdiche, Franck Ravat, Olivier Teste, and Faïez Gargouri. OAEI 2018 results of pomap++. In *OM@ISWC 2018*, pages 192–196, 2018.
16. Amir Laadhar, Faiza Ghozzi, Imen Megdiche, Franck Ravat, Olivier Teste, and Faïez Gargouri. The impact of imbalanced training data on local matching learning of ontologies. In *Business Information Systems - 22nd International Conference, BIS 2019. Proceedings, Part I*, pages 162–175, 2019.

17. Amir Laadhar, Faiza Ghozzi, Imen Megdiche, Franck Ravat, Olivier Teste, and Faïez Gargouri. Pomap++ results for OAEI 2019: Fully automated machine learning approach for ontology matching. In *OM@ISWC 2019*, pages 169–174, 2019.
18. Bing Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data.* Data-centric systems and applications. Springer, Heidelberg ; New York, 2 edition, 2011.
19. Christian Meilicke, Raul Garcia-Castro, Fred Freitas, Willem Robert van Hage, Elena Montiel-Ponsoda, Ryan Ribeiro de Azevedo, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Andrei Tamilin, Cássia Trojahn dos Santos, and Shenghui Wang. Multifarm: A benchmark for multilingual ontology matching. *J. Web Semant.*, 15:62–68, 2012.
20. Christian Meilicke and Heiner Stuckenschmidt. Analyzing mapping extraction approaches. In *Proceedings of the 2nd International Conference on Ontology Matching-Volume 304*, pages 25–36. CEUR-WS. org, 2007.
21. Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Workshop Track Proceedings*, 2013.
22. Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
23. Alexander C Müller and Heiko Paulheim. Towards combining ontology matchers via anomaly detection. In *OM*, pages 40–44, 2015.
24. DuyHoa Ngo, Zohra Bellahsene, and Remi Coletta. A generic approach for combining linguistic and context profile metrics in ontology matching. In *OTM Confederated International Conferences*, pages 800–807. Springer, 2011.
25. Axel-Cyrille Ngonga Ngomo and Michael Röder. Hobbit: Holistic benchmarking for big linked data. *ERCIM News*, 2016(105), 2016.
26. Ikechukwu Nkisi-Orji, Nirmalie Wiratunga, Stewart Massie, Kit-Ying Hui, and Rachel Heaven. Ontology alignment based on word embedding and random forest classification. In *ECML PKDD 2018, Proceedings, Part I*, pages 557–572, 2018.
27. Jan Portisch, Sven Hertling, Heiko Paulheim, A Visual, and Confusion Matrix. Visual analysis of ontology matching results with the melt dashboard. In *The Semantic Web: ESWC 2020 Satellite Events*, 2020.
28. Jan Portisch, Michael Hladik, and Heiko Paulheim. RDF2Vec Light - A Lightweight Approach for Knowledge Graph Embeddings. In *Proceedings of the ISWC 2020 Posters & Demonstrations*, 2020. in print.
29. Jan Portisch and Heiko Paulheim. Alod2vec matcher. In *OM@ISWC 2018*, pages 132–137, 2018.
30. Petar Ristoski, Jessica Rosati, Tommaso Di Noia, Renato De Leone, and Heiko Paulheim. Rdf2vec: RDF graph embeddings and their applications. *Semantic Web*, 10(4):721–752, 2019.
31. Bita Shadgara, Azadeh Haratian Nejhadia, and Alireza Osareha. Ontology alignment using machine learning techniques. *International Journal of Computer Science and Information Technology*, 3, 2011.
32. Lucy Lu Wang, Chandra Bhagavatula, Mark Neumann, Kyle Lo, Chris Wilhelm, and Waleed Ammar. Ontology alignment in the biomedical domain using entity definitions and context. 2018.
33. Stuart N. Wrigley, Raul Garcia-Castro, and Lyndon J. B. Nixon. Semantic evaluation at large scale (SEALS). In *Proceedings of the 21st World Wide Web Conference, WWW 2012*, pages 299–302, 2012.

Learning reference alignments for ontology matching within and across domains *

Beatriz Lima[1], Ruben Branco[2,3], João Castanheira, Gustavo Fonseca[1], and Catia Pesquita[1,3]

[1] LASIGE, Faculdade de Ciências da Universidade de Lisboa, Portugal
[2] NLX—Natural Language and Speech Group, Faculdade de Ciências da Universidade de Lisboa, Portugal
[3] Dep. de Informática, Faculdade de Ciências da Universidade de Lisboa, Portugal
clpesquita@fc.ul.pt

Abstract. Reference alignments are the standard approach for ontology alignment evaluation. However, building a reference alignment is time-consuming and usually depends on expert availability. Several strategies have been proposed to mitigate this issue, ranging from exploring external resources, building simulated alignment tasks, or even crowdsourcing. A simple approach is to take a consensus alignment built from the outputs of several ontology matching systems results.
We present a preliminary investigation that focuses on the generalization of machine learning models trained on the output alignments of multiple systems for a task where a reference alignment is available to other alignment tasks.
Results show that while the consensus alignment works well for alignment tasks where several systems achieve a high performance and produce similar alignments, trained reference models are able to improve on the consensus both within and across domains.

Keywords: Ontology matching · Machine Learning · Reference alignment · OAEI

1 Introduction

The evaluation of ontology alignments typically relies on reference alignments which are automatically compared to the outputs of the alignment systems. Reference alignments are commonly either manually-curated by domain experts or automatically generated. The first kind can be created manually from scratch or manually validated given a set of automatically generated candidates[10]. Although very reliable, they are difficult to obtain as they are very time-consuming and require domain expertise. To decrease the effort and associated cost, both automated strategies and crowdsourcing have been used. Automated strategies, usually work with simulated data [5] or by exploring external resources [9].

Crowdsourcing has been successfully employed, however producing references for complex domains is more difficult to achieve due to the lack of expertise of crowdsourced workers[2]. When the above options are not available, an easy solution to evaluate competing systems is based on a consensus alignment. This strategy is employed by the Disease and Phenotype track at the Ontology Alignment Evaluation Initiative [7] with a consensus alignment built on three votes (i.e., if a mapping is found by 3 different systems it is considered correct). The consensus is considered to be a partial reference alignment and mappings that are generated by a single system are then manually evaluated.

Motivated by the difficulties in generating a reference alignment and inspired by the consensus alignment strategy, we hypothesise that a machine learning model trained on the output alignments of multiple systems for a task where a reference alignment is available, can be used to evaluate other alignment tasks.

2 Methodology

The alignments produced by the ontology matching (OM) tools that participated in the Anatomy, Large BioMed and Conference tracks of OAEI 2019[4] were used as data sources. In our proposed models, each instance corresponds to a mapping in a given alignment task. The features translate in whether the given mapping was present or absent in the output of each of the participating OM tools, taking as values 1 or 0. The reference alignment was used to produce the target class, and support supervised learning. Thus, the model is learning to classify whether a mapping between two ontologies is correct, based on the pattern of outputs of the OM tools while using the reliable reference alignment as ground-truth.

The Anatomy track consists of matching Adult Mouse Anatomy[8] and the portion covering human anatomy of the National Cancer Institute Thesaurus (NCI)[6], and it is supported by a manually-curated reference alignment. Several OM systems achieve a high performance in this track [4]. The Large BioMed track comprises three ontologies, the Foundational Model of Anatomy (FMA)[11], SNOMED CT[3], and NCI. These ontologies are matched pairwise, generating three possible alignments: FMA-NCI, FMA-SNOMED and NCI-SNOMED, which will be further addressed as LB1, LB2 and LB3, respectively, for abbreviation. LB1 and LB2 cover the anatomical domain, whereas LB3 does not. The reference alignment was extracted from an external resource [9]. The Conference track[13] provides 16 ontologies from the conference organisation domain. Since only 7 ontologies are contained in the existing reference alignment, we end up with 21 result alignments, which corresponds to the complete alignment space between these ontologies. We randomly generated 3 different datasets (CF1, CF2, CF3), each of which containing 18 alignments for training and 3 alignments for testing. The alignments used for testing were cmt-ekaw, cmt-conference and iasted-sigkdd in CF1; conference-confof, edas-sigkdd and iasted-sigkdd in CF2; confof-edas, cmt-ekaw and ekaw-sigkdd in CF3.

[4] `http://oaei.ontologymatching.org/2019/`

A reference alignment only contains true positive mappings. Assuming its completeness, every potential mapping that is not a part of the reference is a false mapping. A traditional option to generate negative examples would be a random sampling of entity pairs from each ontology that are not present in the reference alignment. However, this would result in mostly instances with all zero features, and thus uninformative, since most systems produce alignments of cardinality near one. Instead, we take as negative examples all mappings that at least one of the OM tools finds but which are not a part of the reference alignment. To tackle the imbalance caused by this approach, we investigated two different sampling strategies: SMOTE oversampling[1] and undersampling with TomekLinks[12].

Three types of experiments were performed for each domain to verify different properties. In **Experiment 1**, which worked as a baseline, we investigated how well a model can be learned within a given alignment task. We performed 10-fold cross-validation, with a grid search for hyperparameter tuning over a set of 8 machine learning approaches[5]. In **Experiment 2**, we investigated if a model trained in one/more tasks would generalize well to other tasks within the same domain. To support this, features were extracted from the OM tools which participated in both training and test tasks. **Experiment 3** aims to verify how well the method generalises for ontologies in completely different domains. We train on LargeBio data and test on Conference, and vice-versa, again using the intersection of OM tools that participated in both tracks. For all experiments, we also computed the majority vote and the consensus with vote=3 results.

3 Results and Discussion

Table 1 presents the results obtained for all three experiments, using the best sampling strategy (oversampling) and machine learning approaches [6]. In the Biomedical domain, all cross-validation experiments achieved good performance (0.8 to 0.915 average F1-score), however, in the Anatomy task, the *Three votes* approach achieved the best result. In the second experiment, the model learned in Anatomy achieved at best an F1-score of 0.697 in LargeBio, whereas the model trained on LargeBio reached 0.938 in Anatomy. Nevertheless, the *Three votes* consensus approach achieved a higher score in these two cases. However, within the LB track, the ML models outperformed the consensus approach in LB1 and LB2 trained models. These results indicate that system strategies likely differ between the Anatomy and LargeBio tracks. The greater complexity and coverage of LB (which includes both anatomical and non-anatomical tasks) can help explain these results. In the Conference domain, the first experiment results were overall high, with ML approaches improving over the consensus. The second experiment revealed that the ML approaches were able to outperform

[5] Random Forest, K-Nearest Neighbors, Decision Tree, Multi-Layer Perceptron, Naive Bayes, Gradient Boosting, Logistic Regression and Adaboost

[6] The full table of results along with hyperparameter information can be found here: `https://github.com/liseda-lab/ML4ReferenceAlignment`

the consensus in only one test case. As for the cross-domain experiments, we can observe that, even though the LB dataset is much bigger than CF, models trained in CF were able to generalise well to LB and vice-versa, and in both cases surpass the consensus. One relevant aspect that may help explain these results is the agreement degree between OM systems. In the Anatomy task, the average agreement [7] between systems is 0.75, whereas in LB1, LB2, and LB3 it is 0.35, 0.26 and 0.40, respectively. In Conference the agreement ranges between 0.51 and 0.86 with most tasks falling below 0.65. This indicates that when systems have a high agreement, the consensus provides a good evaluation, but when systems differ in their outputs, the ML approaches work best.

Exp.	**Train**	**Test**	**Gradient Boosting**	**AdaBoost**	**Logistic Regression**	**Decision Tree**	Majority Vote	Three votes
				Biomedical				
1	Anatomy		0.915	0.897	0.902	0.909	0.907	**0.945**
	LB1		0.933	**0.935**	0.934	0.923	0.856	0.815
	LB2		**0.885**	0.806	0.822	0.881	0.384	0.689
	LB3		**0.905**	0.901	0.898	0.889	0.771	0.794
2	Anatomy	LB	0.376	0.697	0.629	0.380	0.712	**0.772**
	LB	Anatomy	0.937	0.860	0.147	0.938	0.907	**0.945**
	LB1	LB2+3	**0.794**	0.773	0.775	0.771	0.765	0.688
	LB2	LB1+3	**0.848**	0.807	0.834	0.836	0.786	0.798
	LB3	LB1+2	0.709	0.707	0.711	0.713	0.587	**0.734**
				Conference				
1	CF		**0.803**	0.767	0.770	0.766	0.616	0.668
2	CF1		0.682	**0.789**	0.771	0.651	0.585	0.510
	CF2		0.696	0.689	0.698	0.587	0.677	**0.720**
	CF3		0.571	0.634	0.634	0.609	**0.643**	0.635
				Cross-domain				
3	LB	CF	0.677	**0.678**	0.670	0.674	0.603	0.603
	CF	LB	0.783	0.787	**0.790**	0.788	0.720	0.720

Table 1. F1-scores using oversampling strategy and best classifiers. The values in bold are the best scoring classifiers for each experiment (row).

Our preliminary results highlight an opportunity to address the challenge of incomplete reference alignments by training models with a partial reference. Furthermore, they also showcase that in tasks where systems output dissimilar

[7] computed as the average pairwise jaccard similarity between OM systems outputs

alignments, a model trained in other alignment tasks, even from a different domain, can provide a more complete evaluation than a consensus alignment.

Acknowledgements CP and BL are funded by the FCT through LASIGE Research Unit, ref. UIDB/00408/2020 and ref. UIDP/00408/2020, and by projects SMILAX (ref. PTDC/EEI-ESS/4633/2014). CP is also funded by GADgET (ref. DSAIPA/DS/0022/2018). RB is funded by PORTULAN CLARIN Research Infrastructure through Lisboa 2020, Alentejo 2020 and FCT (PINFRA/22117/2016).

References

1. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: synthetic minority over-sampling technique. Journal of artificial intelligence research **16**, 321–357 (2002)
2. Cheatham, M., Hitzler, P.: Conference v2. 0: An uncertain version of the oaei conference benchmark. In: International Semantic Web Conference. pp. 33–48. Springer (2014)
3. Donnelly, K.: Snomed-ct: The advanced terminology and coding system for ehealth. Studies in health technology and informatics **121**, 279 (2006)
4. Dragisic, Z., Ivanova, V., Li, H., Lambrix, P.: Experiences from the anatomy track in the ontology alignment evaluation initiative. Journal of biomedical semantics **8**(1), 56 (2017)
5. Ferrara, A., Montanelli, S., Noessner, J., Stuckenschmidt, H.: Benchmarking matching applications on the semantic web. In: Extended Semantic Web Conference. pp. 108–122. Springer (2011)
6. Golbeck, J., Fragoso, G., Hartel, F., Hendler, J., Oberthaler, J., Parsia, B.: The national cancer institute's thesaurus and ontology. Journal of Web Semantics First Look 1_1_4 (2003)
7. Harrow, I., Jiménez-Ruiz, E., Splendiani, A., Romacker, M., Woollard, P., Markel, S., Alam-Faruque, Y., Koch, M., Malone, J., Waaler, A.: Matching disease and phenotype ontologies in the ontology alignment evaluation initiative. J Biomed Semantics **8**(1), 55 (2017)
8. Hayamizu, T.F., Mangan, M., Corradi, J.P., Kadin, J.A., Ringwald, M.: The adult mouse anatomical dictionary: a tool for annotating and integrating data. Genome biology **6**(3), 1–8 (2005)
9. Jiménez-Ruiz, E., Grau, B.C., Horrocks, I.: Exploiting the umls metathesaurus in the ontology alignment evaluation initiative.
10. Li, H., Dragisic, Z., Faria, D., Ivanova, V., Jiménez-Ruiz, E., Lambrix, P., Pesquita, C.: User validation in ontology alignment: functional assessment and impact. The Knowledge Engineering Review **34** (2019)
11. Rosse, C., Mejino Jr, J.L.: A reference ontology for biomedical informatics: the foundational model of anatomy. Journal of biomedical informatics **36**(6), 478–500 (2003)
12. Tomek, I.: Two modifications of cnn. IEEE Transactions on Systems, Man, and Cybernetics **SMC-6**(11), 769–772 (1976)
13. Zamazal, O., Svátek, V.: The ten-year ontofarm and its fertilization within the onto-sphere. Journal of Web Semantics **43**, 46–53 (2017)

SUBINTERNM: Optimizing the Matching of Networks of Ontologies

Fabio Santos[1], Kate Revoredo[2], and Fernanda Baião[3]

[1] Northern Arizona University, United States
[2] Vienna University of Economics and Business, Vienna, Austria
[3] Department of Industrial Engineering, Pontifical Catholic University of Rio de Janeiro, Brazil
fd252@nau.edu, kate.revoredo@wu.ac.at, fbaiao@puc-rio.br

Abstract. System of systems (SoS) are interconnected systems that bring value to different domains like health, emergency and crisis management systems. The integration of these SoS creates opportunities to change, validate the information, and add more value to information systems. SoS may have ontologies in their background to support knowledge description and semantic integration. Consequently, the integration of SoSs may benefit from the integration of the network of ontologies behind. However, the task of integrating networks of ontologies, especially the ones describing real-world SoS can be infeasible due to the size of the networks. In this work, we propose an approach, SubInterNM, based on algebraic operations that reduces the number of comparisons needed to match the networks behind the SoSs. We validated our approach using networks of ontologies created from the OAEI ontologies. The SubInterNM combined with Alin and LogMap can overcome these matchers, when running alone in some cases. *

1 Introduction

A System-of-Systems (SoS) is defined as a set of independent systems, providing functionalities derived from the interoperability among them [2]. Examples of SoS scenarios are smart cities, health, and emergency response systems, and crisis management systems [5]. Current SoS are increasingly supported by networks of ontologies, which provide a semantic backbone for modeling and reasoning over data.

A network of ontologies (NO) is a set of two or more aligned ontologies. The network can represent a set of domains of their compound ontologies. Each ontology describes the knowledge of a domain of interest [9]. Many ontologies networks may be created inside organizations as a response to the demand for a semantic interoperability layer among their information systems (IS).

In current scenarios demanding data integration and systems interoperability (such as company acquisitions) within the same business domain, different SoS must be integrated. Because of the intrinsically multiple possible relationships inside the SoS that includes IS from distinct companies, the integration can be challenging and may be viewed as a matching of ontologies networks, requiring mapping concepts between both SoSs. This research addresses how to match a network of ontologies.

This study proposes an approach to optimize internetwork matching in the context of networks of ontologies, by systematically examining the characteristics of the NO and avoiding computing all possible matchings between entities. More specifically, we implement the subsumed internetwork matching (SubInterNM) [8], which reduces the number of pairs to be evaluated in the matching process. We evaluated the proposed approach in a preliminary experiment using an OAEI dataset.

This work addresses the following research question: *"Is it possible to align two ontology networks without computing all the possible alignments, with viable computational effort, time and precision, recall and f-measure?"*. To answer this question, this paper contributes by proposing an approach to match network of ontologies and its implementation into a prototype tool, as well as an empirical study showing the viability of the approach and its performance gains over state-of-the-art matching tools.

This work is organized as follows: in section 2, we summarize background. Evaluation results and a discussion are presented in section 3. Section 4 describes the limitations, and finally, section 5 concludes and points to future work.

2 Background

Current approaches for ontology matching include pairwise matching [9] and holistic matching [7]. Both may be adapted to match Networks of Ontologies; however, they perform an exhaustive checking of every single possible pair of entities for each ontology that composes the networks. They also have limited scalability, since the required number of steps for computing all the alignments grows exponentially to the number and sizes of the ontologies composing each network. Indeed, both pairwise or holistic approaches are not prepared to match Networks of Ontologies [8] since they do not take into account the structure of the networks and cannot limit the number of comparisons.

The ontology matching problem is not new and has been researched for a decade. However, to our knowledge, the matching of Network of Ontologies has not received the same attention. Although there are few studies dealing with networks of ontologies and, consequently, matching networks, there are still open challenges [1].

3 Results and Discussion

To assess SubInterNM we conducted an experiment using ontologies from the OAEI conference domain, so as to limit the size and help checking the results manually.

We selected Alin [4] and LogMap [6] as the matchers for the experiment. Alin obtained one of the best metrics in the OAEI initiative, and LogMap is one of the best to handle large ontologies. The SubInterNM approach uses the definitions in [3] and is available online [b]. For the sake of reprodutibility, the results are also available online [c].

We first selected ontologies to compose two networks of ontologies. Since LogMap and Alin are not able to natively process networks, the baseline submitted as input to

[b] https://github.com/fabiojavamarcos/interNetworkOntologyMatching

[c] zenodo.org (10.5281/zenodo.3977855)

Table 1. Precision Recall and F-Measure subInterNM+LogMap(1) subInterNM+Alin(2) LogMap(3) Alin(4) - *Did not calculate the metrics

Exp	P(1)	R(1)	F(1)	P(2)	R(2)	F(2)	P(3)	R(3)	F(3)	P(4)	R(4)	F(4)
2x2	0.818	0.600	0.692	**0.90**	**0.667**	**0.769**	0.842	0.432	0.571	0.348	0.405	0.375
4x4	0.818	0.600	0.692	**0.90**	**0.667**	**0.769**	0.750	0.141	0.237	0.197	0.192	0.195
5x5	0.818	0.600	0.692	**0.90**	**0.667**	**0.769**	0.583	0.126	0.207	0.096	0.102	0.099
5x1	0.690	0.233	0.348	**0.904**	0.221	0.355	0.778	**0.244**	**0.371**	0.0*	0.0*	0.0*
5x2	0.655	**0.268**	**0.380**	**0.888**	0.225	0.359	0.388	0.164	0.265	0.537	0.132	0.212
5x3	0.833	**0.435**	**0.572**	**0.882**	0.326	0.476	0.674	0.179	0.283	0.3	0.162	0.21

these matchers consisted of the union of all ontologies from each network. The alternative scenario to be compared consisted of executing SubInterNM and then submitting its partial results to the LogMap and Alin matchers. Following, each possible pair of ontologies was submitted to LogMap and Alin alone, with duplication and without duplication (when pairs of the same ontologies were manually eliminated, i.e. Edas x Ekaw and Ekaw x Edas; Edas x Edas). For each scenario we collected the following metrics: processing time, average precision, average recall, and average f-measure.

- 2x2: $\Omega = \{$sigkdd, confof$\}$ and $\Omega' = \{$conference, confof$\}$;
- 4x4: $\Omega = \{$sigkdd, confof, ekaw, edas $\}$ and $\Omega' = \{$conference, confof, ekaw, edas$\}$;
- 5x5: $\Omega = \{$sigkdd, confof, ekaw, edas, iasted $\}$ and $\Omega' = \{$conference, confof, ekaw, edas, iasted$\}$;
- 5x1: $\Omega = \{$sigkdd, confof, ekaw, edas, iasted$\}$ and $\Omega' = \{$conference$\}$;
- 5x2: $\Omega = \{$sigkdd, confof, ekaw, edas, iasted $\}$ and $\Omega' = \{$conference, confof$\}$;
- 5x3: $\Omega = \{$sigkdd, confof, ekaw, edas, iasted$\}$ and $\Omega' = \{$conference, confof, ekaw$\}$;

The results show higher precision, recall, and f-measure using the SubInterNM combined with the Alin compared with the matcher alone and when combined with LogMap (Table 1). LogMap combined with SubInterNM or alone was faster than SubInterNm+Alin or Alin alone, even when the network size grew (Table 2).

The metrics (Table 1) showed a decrease in the values of the Alin and LogMap approach as the network grew. It can be explained by the lack of flexibility of the solutions in understanding a reference alignment that contains concepts coming from different ontologies. It occurs because when finishing the union operation, the resulting temporary ontology is composed of the union of concepts from all the ontologies together. LogMap handled better than complexity and loosed significantly less precision than Alin. The experiments using the matcher alone started aligning the structure after the union to standardize the input for all approaches.

In Table 3, we ran the matcher in all possible combinations, O_i x O_j, (column "All") or without duplications (column "Time"). We computed the sum of processing time and the average of the quality metrics. Alin obtained higher quality metrics again, while LogMap continued to be faster. Looking at the 5x5 and 5x1 cases, we observe that Alin used significantly more time than when combined with the SubInterNM, but delivered better quality metrics. LogMap outperformed all the options wrt processing time and obtained similar metrics in the 5x5 case and better ones in the 5x1 case.

Table 2. Processing Time (seg)

Experimemt	subInterNM+LogMap	subInterNM+Alin	LogMap	Alin
2x2	4.345	8.587	**3.766**	8.71
4x4	29.736	34.931	**20.541**	22.516
5x5	65.65	71.494	**19.999**	32.137
5x1	16.548	21.442	**11.668**	20.537
5x2	20.041	25.151	**11.4**	15.989
5x3	23.496	29.445	**11.784**	19.445

Examining the column "All" (Table 3), we observe that LogMap processing times were comparable to SubInterNM + LogMap, but the latter approach produced a result without duplicated alignments. Alin alone had better results than SubInterNM + Alin but used significantly more time and delivered solutions with duplications and may cost more $O(nlogn)$ effort and more time. In networks with many isomorphisms, the SubInterNM + Alin delivered more balanced results combining metrics and time processing. On the other hand, classic pairwise approaches have better outcomes when the networks had few isomorphisms.

Finally, it is possible to avoid the Cartesian product and keep processing time and resulting metrics depending on the characteristics of the networks being aligned.

Table 3. Time (seg) and average (Precision, Recall and F-Measure) - Pairwise Individually

Exp	Alin Time	Alin All	P	R	F	LogMap Time	LogMap All	P	R	F
2x2	18.581	24.292	**0.950**	**0.889**	**0.918**	**8.323**	**10.274**	0.897	0.727	0.801
4x4	63.892	88.148	**0.942**	**0.756**	**0.836**	**25.447**	**33.615**	0.809	0.604	0.688
5x5	96.768	163.343	**0.979**	**0.721**	**0.823**	**38.521**	**63.556**	0.853	0.592	0.674
5x1	32.570	32.570	**0.934**	**0.74**	**0.820**	**15.256**	**15.256**	0,753	0.595	0.660
5x2	57.076	62.787	**0.972**	**0.768**	**0.853**	**24.076**	**26.027**	0.891	0.596	0.707
5x3	82.548	94.621	**0.959**	**0.711**	**0.815**	**33.368**	**37.238**	0.845	0.631	0.719

4 Limitations

When pruning the Network of Ontologies to reduce the posterior matcher computations, some entities can be missing. This may impact on how the similarity algorithms find the alignments, which may lead to different results. The use of a dataset from the same domain is not a real scenario, as discussed in Section 1. Yet, this enabled us to manually verify the algebraic operations and the computed metrics, since we needed many customized reference alignments.

Because of the many possibilities in the experiments, we needed to create some new reference alignments based on the existing from OAEI. These were validated by the research group but are not error-proof.

Finally, the intrinsic characteristics of the ontologies considered in the experiment generated sparse graphs, which may have helped the algebraic algorithms. In scenarios with more dense ontologies (i.e., with more connections among their concepts), we could have strongly connected graphs and, consequently, worse time processing when using our proposed SubInterNM.

5 Future Work and Conclusion

This paper contributes by presenting a novel concept of a Network of Ontologies Matcher approach.The proposal was implemented in a prototype to show its feasibility and confirm (our research question) that it is possible to align two ontology networks without computing all the possible alignments, with viable computational effort, time and precision, recall and f-measure. As predicted, the experiment results confirmed that SubInterNM computed the matching among distinct networks of ontologies more efficiently than using the traditional pairwise approach in specific cases, due to avoiding unnecessary comparisons without losing information.

For future work, we aim to run experiments with larger ontologies and networks, discover the optimal point where the SubInterNM can be used and add a strategy to retracting/forgetting axioms/entities [10] while preserving entailment.

References

1. de Abreu Santos, F.M., Revoredo, K., Baião, F.A.: Paving a research roadmap on network of ontologies. In: OM@ ISWC. pp. 221–222 (2017)
2. Boehm, B.: A view of 20th and 21st century software engineering. In: Proceedings of the 28th international conference on Software engineering. pp. 12–29. ACM (2006)
3. Casanova, M.A., Magalhães, R.C.: Operations over lightweight ontologies and their implementation. In: Implicit and Explicit Semantics Integration in Proof-Based Developments of Discrete Systems, pp. 61–82. Springer (2020)
4. Da Silva, J., Revoredo, K., Baião, F., Euzenat, J.: A lin: improving interactive ontology matching by interactively revising mapping suggestions. The Knowledge Engineering Review **35** (2020)
5. Fitzgerald, J., Foster, S., Ingram, C., Larsen, P.G., Woodcock, J.: Model-based engineering for systems of systems: the compass manifesto. COMPASS Interest Group, Tech. Rep. Manifesto Version **1** (2013)
6. Jiménez-Ruiz, E., Cuenca Grau, B.: Logmap: Logic-based and scalable ontology matching. In: International Semantic Web Conference. pp. 273–288. Springer (2011)
7. Megdiche, I., Teste, O., Trojahn, C.: An extensible linear approach for holistic ontology matching. In: International Semantic Web Conference. pp. 393–410. Springer (2016)
8. Santos, F., Revoredo, K., Baiao, F.: A proposal for optimizing internetwork matching of ontologies. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop. p. 71 (2018)
9. Shvaiko, P., Euzenat, J.: Ontology matching: state of the art and future challenges. IEEE Transactions on knowledge and data engineering **25**(1), 158–176 (2013)
10. Wang, K., Wang, Z., Topor, R.W., Pan, J.Z., Antoniou, G.: Concept and role forgetting in *ALC*{\mathcal {ALC}} ontologies. In: The Semantic Web - ISWC 2009, 8th International Semantic Web Conference. pp. 666–681

A Survey of OpenRefine Reconciliation Services

Antonin Delpeuch[1] [0000−0002−8612−8827]

Department of Computer Science, University of Oxford, UK
antonin.delpeuch@cs.ox.ac.uk

Abstract. We give an overview of the OpenRefine reconciliation API, a web protocol for tabular data matching. We suggest that such a protocol could be useful to the ontology matching community to evaluate systems more easily, following the success of the NIF ontology in natural language processing. This would make it easier for linked open data practitioners to build on the systems developed for evaluation campaigns. The OAEI task formats suggest some changes to the protocol specifications.

Keywords: record linkage · entity matching · reconciliation service · deduplication · web standards

1 Introduction

Integrating data from sources which do not share common unique identifiers often requires matching (or *reconciling*, *merging*) records which refer to the same entities. This problem has been extensively studied and many heuristics have been proposed to tackle it [1]. The Ontology Alignment Evaluation Initiative runs a yearly competition on this topic, offering a variety of task formats.

The OpenRefine reconciliation API[1] is a web protocol designed for this task. While most software packages for record linkage assume that the entire data is available locally and can be indexed and queried at will, this protocol proposes a workflow for the case where one of the data souces to be matched is held in an online database. By implementing such an interface, the online database lets users match their own datasets to the identifiers it holds. The W3C Entity Reconciliation Community Group[2], has been formed to improve and promote this protocol.

In this article, we survey the existing uses of the protocol and propose an architecture based on it to run evaluation campaigns in ontology matching.

[1] https://reconciliation-api.github.io/specs/latest/

[2] https://www.w3.org/community/reconciliation/

```
{
  "query": "Cesaria Evora",
  "type": "DifferentiatedPerson",
  "properties": [
    {
      "pid": "dateOfBirth",
       "v": "1941-08-27"
    }
  ]
}
```

(a) A reconciliation query

```
[
 {
  "id": "121291081",
  "name": "Évora, Cesária",
  "score": 92.627655,
  "match": true,
  "type":[
     {"id": "AuthorityResource"},
     {"id": "DifferentiatedPerson"}]
 },
 ...
]
```

(b) Response with candidates entities

Fig. 1: Example of a reconciliation workflow

2 Overview of the reconciliation protocol

The reconciliation API is essentially a search protocol tailored to the reconciliation problem. This protocol is implemented by many servers[3] and clients[4]. Consider the query in Figure 1. It contains the following components:

- The name of the entity to search for;
- An optional type to which the search should be restricted. The possible types are defined by the reconciliation service itself;
- An optional array of property values to refine the matching. The ontology is also defined by the reconciliation service.

We can submit this query to the reconciliation endpoint `https://lobid.org/gnd/reconcile`, which exposes the authority file of the German National Library (GND). As a response, we get a list of candidates ranked by score and a matching decision, predicting whether the entity matches the query.

The canonical client for this API is OpenRefine[5] [4], a data cleaning tool which can be used to transform raw tabular data into linked data. The tool proposes a semi-automatic approach to reconciliation, making it possible for the user to review the quality of the reconciliation candidates returned by the service. To that end, the reconcilation API lets services expose auto-complete endpoints and HTML previews for the entities they store, easing integration in the user interface of the client.

[3] A list of publicly available endpoints can be found at `https://reconciliation-api.github.io/testbench/`

[4] `https://reconciliation-api.github.io/census/clients/`

[5] `http://openrefine.org/`

3 Potential use in OAEI evaluation campaigns

In this section we turn our attention to the Ontology Alignment Evaluation Initiative, whose tasks cover among others the alignment of tabular data to knowledge bases. In these campaigns, reconciliation heuristics are evaluated on datasets covering various topics. Participants submit their systems which are run by evaluation platforms on test datasets, and their results are compared to reference alignments provided by the organizers. We argue that a web-based API such as the reconciliation API would be useful in OAEI campaigns, for multiple reasons.

The evaluation of candidate systems in OAEI events is carried out using various platforms. SEALS [8] is a Java-based tool to evaluate matching systems which has been used in OAEI campaigns for about 10 years. To be compatible with SEALS, matching systems must implement a Java interface which offers an API for ontology alignment. Participants who want to develop their systems in other programming languages have to write a Java wrapper around them, in order to be compatible with the evaluator. More recently, the HOBBIT [6] platform proposed a similar approach, where systems are submitted as Docker images and communicate with the evaluator in a similar way. Finally, the MELT platform [3] was proposed this year as a Java framework to develop systems compatible with both HOBBIT and SEALS. The newly launched SemTab challenge has been using the AIcrowd[6] platform so far. This platform does not evaluate systems directly, as participants submit the alignments produced by their systems on their own.

The complexity of this ecosystem is daunting for new participants. It also unlikely that systems packaged for the OAEI challenges are reused as such outside academia, for instance by an investigative journalist who would like to match company names to records in company registers or by a linked data enthusiast who would like to import a dataset in Wikidata.

We argue here that the communication between the evaluator and participating systems could be done via a web protocol such as the reconciliation API. This architecture is already been used in other domains. For instance, in natural language processing, it is used for *entity linking* (annotating text with mentions of named entities aligned to a knowledge base). The GERBIL platform [7] evaluates systems for this task using a web API based on NIF [2], an ontology to represent text annotation tasks. Experiments can be configured from a web interface, letting the user choose systems, datasets and evaluation metrics. Experiment results are then archived publicly.

The use of a web-based architecture has three main benefits. First, academics can evaluate their entity linking system simply by submitting to GERBIL the URL of their service. They can easily compare their systems to other services available online. Debugging services on some input data can be done easily with

[6] `https://www.aicrowd.com/challenges/semtab-2020-cell-entity-annotation-cea-challenge`

a web browser.[7] Second, systems can be used outside academia easily, as users only need to interact with a simple web API without installing anything. In turn, this use of the systems by practitioners can help source new datasets for evaluation campaigns. For instance, the Wikidata reconciliation service serves millions of queries each month. These queries can be logged, analyzed and turned into new datasets which match real-world use cases closely.

4 Adapting the protocol to the OAEI tasks

The protocol specifications are actively being discussed and improved with feedback from users, service providers and other stakeholders. Therefore, if we identify aspects of the protocol which do not fit well with the use case sketched above, it is possible to address them in a new version of the specifications.

In the SemTab challenge, the task is to match table cells to entities of a knowledge graph, without any information about the relations between columns or the domain of the dataset: these must be inferred by the service too. In contrast, reconciliation queries already identify the role of each data field using the service's ontology. One could therefore wonder whether the reconciliation protocol should be adapted not to require this information.

The anonymous reviewers have also been helpful in pointing out points that we have then forwarded to the Community Group. For instance, in some tasks a given cell can be matched to multiple entities[8]. Another useful comment was made about the absence of multilingual support in the API,[9] which had also been brought up in a different context.

5 Conclusion

We have surveyed a range of services which conform to the reconciliation API. The use of a web API such as the reconciliation API could well benefit academic initiatives such as OAEI, especially for the newly-lauched challenge on alignment of tabular data to knowledge bases [5]. Therefore, we hope to see fruitful interactions between these two communities in the future. We encourage all interested parties to join the W3C Entity Reconciliation Community Group[10].

6 Acknowledgements

We thank the anonymous reviewers, the OpenCorporates team, Vladimir Alexiev and the W3C Entity Reconciliation Community Group for their feedback on this project. This work was supported by OpenCorporates as part of the

[7] The reconciliation testbench can be used to submit queries to services: `https://reconciliation-api.github.io/testbench/`

[8] `https://github.com/reconciliation-api/specs/issues/51`

[9] `https://github.com/reconciliation-api/specs/issues/52`

[10] `https://www.w3.org/community/reconciliation/`

"TheyBuyForYou" project on EU procurement data. This project has received funding from the European Commission's Horizon 2020 research and innovation programme (grant agreement n 780247).

References

1. Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. Springer Science & Business Media (2012)
2. Hellmann, S., Lehmann, J., Auer, S., Brümmer, M.: Integrating NLP Using Linked Data. In: Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Salinesi, C., Norrie, M.C., Pastor, Ó. (eds.) Advanced Information Systems Engineering, vol. 7908, pp. 98–113. Springer Berlin Heidelberg, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41338-4$_7$
3. Hertling, S., Portisch, J., Paulheim, H.: MELT - Matching EvaLuation Toolkit. In: Acosta, M., Cudré-Mauroux, P., Maleshkova, M., Pellegrini, T., Sack, H., Sure-Vetter, Y. (eds.) Semantic Systems. The Power of AI and Knowledge Graphs, vol. 11702, pp. 231–245. Springer International Publishing, Cham (2019). https://doi.org/10.1007/978-3-030-33220-4$_1$7
4. Huynh, D., Morris, T., Mazzocchi, S., Sproat, I., Magdinier, M., Guidry, T., Castagnetto, J.M., Home, J., Johnson-Roberson, C., Moffat, W., Moyano, P., Leoni, D., Peilonghui, Alvarez, R., Vishal Talwar, Wiedemann, S., Verlic, M., Delpeuch, A., Shixiong Zhu, Pritchard, C., Sardesai, A., Thomas, G., Berthereau, D., Kohn, A.: OpenRefine (2019). https://doi.org/10.5281/zenodo.595996
5. Jimenez-Ruiz, E., Hassanzadeh, O., Efthymiou, V., Chen, J., Srinivas, K.: SemTab 2019: Resources to Benchmark Tabular Data to Knowledge Graph Matching Systems. In: The Semantic Web. pp. 514–530. Springer, Cham (May 2020). https://doi.org/10.1007/978-3-030-49461-2$_3$0
6. Ngomo, A.C.N., Röder, M.: HOBBIT: Holistic Benchmarking for Big Linked Data p. 2
7. Usbeck, R., Eickmann, B., Ferragina, P., Lemke, C., Moro, A., Navigli, R., Piccinno, F., Rizzo, G., Sack, H., Speck, R., Troncy, R., Röder, M., Waitelonis, J., Wesemann, L., Ngonga Ngomo, A.C., Baron, C., Both, A., Brümmer, M., Ceccarelli, D., Cornolti, M., Cherix, D.: GERBIL: General Entity Annotator Benchmarking Framework. In: Proceedings of the 24th International Conference on World Wide Web - WWW '15. pp. 1133–1143. ACM Press, Florence, Italy (2015). https://doi.org/10.1145/2736277.2741626
8. Wrigley, S.N., García-Castro, R., Nixon, L.: Semantic evaluation at large scale (SEALS). In: Proceedings of the 21st International Conference Companion on World Wide Web - WWW '12 Companion. p. 299. ACM Press, Lyon, France (2012). https://doi.org/10.1145/2187980.2188033

LIGER – Link Discovery with Partial Recall

Kleanthi Georgala[1,2], Mohamed Ahmed Sherif[2], and Axel-Cyrille Ngonga Ngomo[1,2]

[1] Department of Computer Science, Paderborn University, Germany,
[2] Department of Computer Science, University of Leipzig, Germany
georgala@informatik.uni-leipzig.de
{mohamed.sherif,axel.ngonga}@upb.de

Abstract. In this work, we present a novel approach for link discovery under constraints pertaining to the expected recall of a link discovery task. Given a link specification, the approach aims to find a subsumed link specification that achieves a lower run time than the input specification while abiding by a predefined constraint on the expected recall it has to achieve. Our approach, combines downward refinement operators with monotonicity assumptions to detect such specifications. Our results suggest that our different implementations can detect subsumed specifications that abide by expected recall constraints efficiently, thus leading to significantly shorter overall run times than our baseline.

1 Introduction

Sensor data is used in a plethora of modern Industry-4.0 applications such as condition monitoring and predictive maintenance. An increasing number of such machines generate knowledge graphs in the Resource Description Framework (RDF) format. A key step for learning axioms which generalize well is to learn them across several machines. However, single machines generate independent data streams. Hence, time-efficient data integration (in particular link discovery, short LD) approaches must precede the machine learning approaches to integrate data streams from several machines. Given that new data batches are available periodically (e.g., every 2 hours), practical applications of machine learning on RDF streams demand LD solutions which can guarantee the completion of their computation under constraints such as time (i.e., their total runtime for a particular integration task) or expected recall (i.e., the estimated fraction of a given LD task they are guaranteed to complete).

In this paper, we address the problem of LD with partial recall by proposing LIGER, the first *partial-recall LD approach*. Given a link specification L that is to be executed, LIGER aims to compute a portion of the links returned by L efficiently, while achieving a guaranteed expected recall. LIGER relies on a refinement operator, which allows the efficient exploration of potential solutions to this problem. The main contributions of our work are: (1) We present a downward refinement operator that allows the detection of subsumed LSs with partial recall. (2) We use a monotonicity assumption to improve the time efficiency of our approach. (3) We evaluate LIGER using benchmark datasets.

 This work has been supported by the EU H2020 project KnowGraphs (GA no. 860801) as well as the BMVI projects LIMBO (GA no. 19F2029C) and OPAL (GA no. 19F2028A).

2 Linking with Guaranteed Expected Recall

A knowledge base K is a set of triples $(s, p, o) \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{I} \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L})$, where $\mathcal{I}$ is the set of all Internationalized Resource Identifiers (IRIs) $\mathcal{B}$ is the set of all RDF blank nodes and $\mathcal{L}$ is the set of all literals. Given two sets of RDF resources S and T from two (not necessarily distinct) knowledge bases as well as a relation R, the main goal of LD is to discover the *mapping* $\mu = \{(s, t) \in S \times T : R(s, t)\}$. To achieve this goal, declarative LD frameworks rely on link specifications (LSs), which describe the conditions under which $R(s, t)$ can be assumed to hold for pairs $(s, t) \in S \times T$. Several grammars have been used for describing a LS in previous works [3]. In general, these grammars assume that a LS consists of : (i) *Similarity measures* m ($m : S \times T \times \mathcal{P}^2 \rightarrow [0, 1]$), through which property values of resources found in the input datasets S and T can be compared and (ii) *operators op*. An *Atomic LS* is a filter $f = (m, \tau)$, where m is a similarity measure and $\tau \in [0, 1]$ is a threshold. *Operators* combine two LSs L_1 and L_2 to a more complex specification $L = (f, \tau, op(L_1, L_2))$. For $L = (f, \tau, op(L_1, L_2))$, we call *op the operator* of L. We define the mapping $[[L]] \subseteq S \times T$ of the LS L as the set of links that will be computed by L when applied to $S \times T$. We denote the size of a mapping $[[L]]$ by $|[[L]]|$. We define the *selectivity* of a link specification L as $sel(L) = |[[L]]|/|S \times T|$. The aim of sel is to encode the predicted value of $|[[L]]|$ as a fraction of $|S \times T|$. This is akin to the selectivity definition often used in the database literature. A LS L' is said to achieve a recall k w.r.t. to L if $k \times |[[L]]| = |[[L]] \cap [[L']]|$. If $[[L']] \subseteq [[L]]$, then the recall k of L' abides by the simpler equation $k \times |[[L]]| = |[[L']]|$. A specification L' with $[[L']] \subseteq [[L]]$ is said to achieve an expected recall k w.r.t. to L if $k \times sel(L) = sel(L')$. Formally, given a specification L, the aim of partial-recall LD is to detect a rapidly executable LS $L' \sqsubseteq L$ with an expected recall of at least $k \in [0, 1]$, i.e. a LS L' with $sel([[L']]) \geq k \times sel([[L]])$, where $k \in [0, 1]$ is a minimal expected recall set by the user.

The LS L' is subsumed by the LS L (denoted $L \sqsubseteq L'$) when $[[L]] \subseteq [[L']]$ for any fixed pair of sets S and T. Note that $\sqsubseteq$ is a quasi-ordering (i.e., reflexive and transitive) on the set of all LS, which we denote $\mathcal{LS}$. A key observation that underlies our approach is as follows: $\forall \theta, \theta' \in [0, 1]\ \theta > \theta' \rightarrow (m, \theta) \sqsubseteq (m, \theta')$. This observation can be extended to LS as follows: (1) $L_1 \sqsubseteq L_1' \rightarrow (L_1 \sqcup L_2) \sqsubseteq (L_1' \sqcup L_2)$, (2) $L_1 \sqsubseteq L_1' \rightarrow (L_1 \sqcap L_2) \sqsubseteq (L_1' \sqcap L_2)$, (3) $L_1 \sqsubseteq L_1' \rightarrow (L_1 \backslash L_2) \sqsubseteq (L_1' \backslash L_2)$, and (4) $L_2 \sqsubseteq L_2' \rightarrow (L_1 \backslash L_2') \sqsubseteq (L_1 \backslash L_2)$

We call $\rho : \mathcal{LS} \rightarrow 2^{\mathcal{LS}}$ a *downward refinement operator* if $\forall L \in \mathcal{LS} : L' \in \rho(L) \rightarrow L' \sqsubseteq L$, where $(\mathcal{LS}, \sqsubseteq)$ is a quasi-ordered space. L' is called a *specialisation* of L. We denote $L' \in \rho(L)$ with $L \rightsquigarrow_\rho L'$. Given L as input, the idea behind our approach is to use a refinement operator to compute $L' \sqsubseteq L$ with at least a given expected recall k w.r.t L. We define the corresponding refinement operator over the space $(2^{\mathcal{LS}}, \sqsubseteq)$ as follows:

$$\rho(L) = \begin{cases} \emptyset & \text{if } L = L_\emptyset, \\ L_\emptyset & \text{if } L = (m, 1), \\ (m, next(\theta)) & \text{if } L = (m, \theta) \wedge \theta < 1, \\ (\rho(L_1) \sqcup L_2) \cup (L_1 \sqcup \rho(L_2)) & \text{if } L = L_1 \sqcup L_2, \\ (\rho(L_1) \sqcap L_2) \cup (L_1 \sqcap \rho(L_2)) & \text{if } L = L_1 \sqcap L_2, \\ \rho(L_1) \backslash L_2 & \text{if } L = L_1 \backslash L_2. \end{cases} \tag{1}$$

Our refinement operator ρ is finite, incomplete, proper and redundant if L, S and T are finite. ρ being incomplete is not a restriction for our purposes given that we aim to find Ls that run faster and thus do not want to refine the input LS L to L' that might make our implementation of the operator slower. Given that ρ is *finite*, we can generate ρ for any chosen node completely in our implementation. ρ *being redundant* means that after a refinement, we need to check whether we have already seen the newly generated LS. Hence, we need to keep a set of seen LS. Finally, ρ *being proper* means that while checking for redundancy, there is no need to compare LS with any of their parents.

3 Approach

The basic goal behind LIGER is to find the LS $\Lambda \in \rho^*(L_0)$ that achieves the lowest expected run time while (1) being subsumed by L_0 and (2) achieving at least a predefined excepted recall $k \in [0, 1]$. LIGER is based on ρ as described in Section 2.

The basic implementation of LIGER is dubbed C-RO. Our approach takes as input: a LS L_0, an oracle O which can predict the run times and selectivity of LS, the minimal expected recall k and a refinement time constraint *maxOpt*. We begin by asking O to provide the algorithm with estimations of the selectivity of L_0 (sel_{L_0}). We define a refinement tree with L_0 as its root. For each refined LS, the set of refined LSs are added as children nodes to the currently refined LS, and a leaf node is as a LS that can not be refined any further. We assign L_0 as the best subsumed LS Λ and the best run time rt_Λ with L_0's runtime estimation from O. The algorithm computes the desired selectivity value (sel_{des}) as a fraction of L_0's selectivity. Then, we add L_0 to the set $Buffer$, that serves as a buffer and includes LSs obtained by refining L_0 that have not yet been refined. All LSs that were generated through the refinement procedure as well as the input LS L_0 are stored in another buffer named $Total$. By keeping track of these LSs, we avoid refining a LS more than once and address the redundancy of our refinement operator. The refinement of L_0 stops when the refinement time has exceeded *maxOpt*, or if the $Buffer$ is empty or the selectivity of Λ returned by O is equal to sel_{des}. At each iteration, the algorithm selects the next node for refinement as follows: first, it retrieves the run time estimation of each LS L that belongs to $Buffer$ using O. Then, it selects the next LS for refinement as the LS with minimum run time estimation (L_{xt}). The algorithm then checks if L_{xt} receives a better runtime score, and assigns L_{xt} as the new value of Λ and updates rt_Λ accordingly. Finally, the algorithm refines L_{xt} by implementing ρ. For each subsumed LS, the algorithm checks if it already exists in set $Total$, to ensure that LIGER does not explore LSs that have already been seen before. Each remaining subsumed LS is added to $Total$ and the algorithm proceeds in computing its selectivity. If its selectivity is higher or equal to the desired selectivity, the algorithm updates $Buffer$ by adding the new LS.

One key observation pertaining to the run time of $L' \in \rho^*(L)$ is that by virtue of $L' \sqsubseteq L$, $RT(L') \leq RT(L)$ will most probably hold. By virtue of the transitivity of $\leq$, $L_1 \in \rho(L) \wedge L_2 \in \rho(L) \wedge RT(L_1) \leq RT(L_2) \rightarrow \forall L' \in \rho^*(L_1)$: $RT(L') \leq RT(L_2)$ also holds. We call this assumption the *monotonicity of run times*. Since the implementation of C-RO does not take this monotonicity into consideration, we wanted to know whether this assumption can potentially improve the run time of our approach. We then

Table 1: Average execution times of *Baseline*, C-RO and RO-MA for $k = 0.1$ and different values of *maxOpt* over 100 LS per dataset. All times are in seconds.

$k = 0.1$

	Abt-Buy			Amazon-GP			DBLP-ACM			DBLP-Scholar		
maxOpt	**Baseline**	**C-RO**	**RO-MA**	**Baseline**	**C-RO**	**RO-MA**	**Baseline**	**C-RO**	**RO-MA**	**Baseline**	**C-RO**	**RO-MA**
0.1	0.66	0.52	0.52	5.71	3.91	3.82	1.08	0.25	0.25	792.81	596.53	598.44
0.2	0.66	0.55	0.54	5.71	3.81	2.89	1.08	0.26	0.26	792.81	545.22	546.01
0.4	0.66	0.45	0.44	5.71	3.04	2.91	1.08	0.26	0.25	792.81	589.72	587.87
0.8	0.66	0.55	0.53	5.71	3.27	3.15	1.08	0.28	0.26	792.81	598.54	599.03
1.6	0.66	0.54	0.51	5.71	3.47	3.18	1.08	0.33	0.28	792.81	554.82	557.06

	MOVIES			TOWNS			VILLAGES		
maxOpt	**Baseline**	**C-RO**	**RO-MA**	**Baseline**	**C-RO**	**RO-MA**	**Baseline**	**C-RO**	**RO-MA**
0.1	4.05	1.89	1.89	44.52	31.15	31.20	123.58	15.32	15.26
0.2	4.05	1.75	1.76	44.52	32.23	32.19	123.58	15.65	15.71
0.4	4.05	1.91	1.90	44.52	34.21	34.09	123.58	14.10	14.12
0.8	4.05	1.77	1.76	44.52	34.08	34.10	123.58	14.69	14.52
1.6	4.05	1.93	1.89	44.52	34.38	34.00	123.58	15.65	15.17

implemented an extension of LIGER with the monotonicity assumption (dubbed RO-MA). RO-MA uses a hierarchical ordering on the set of unrefined nodes. By incorporating RO-MA as a search strategy, the refinement tree is expanded using a "top-down" approach until there are no nodes to be further explored in a particular path.

4 Evaluation

We evaluated our approach on seven datasets [4, 2]. All LS used during our experiments were generated automatically by the unsupervised version of the genetic-programming-based ML approach EAGLE [5] as implemented in LIMES [6]. Regarding the Oracle O mentioned in Section 3, LIGER assumes that it can (i) approximate its run time using a linear model described in [1], and (ii) estimate its selectivity as follows: (1) For an atomic LS, the selectivity values were computed using $\frac{|[[L]]|}{|S|\times|T|}$, where $|[[L]]|$ is the size of the mapping returned by the LS L, $|S|$ and $|T|$ are the sizes of the source and target data. To do so, we pre-computed the real selectivity of atomic LSs that were based on a set of measures using the methodology presented in [7] for thresholds between 0.1 and 1. (2) For complex LSs, which are binary combinations of two LSs L_1 (selectivity: $sel(L_1)$) and L_2 (selectivity: $sel(L_2)$), the run time approximation was computed by summing up the individual run times of L_1,L_2. Therefore, we derived the following selectivities: (I) $op(L) = \cap \rightarrow sel(L) = \frac{1}{2}sel(L_1)sel(L_2)$, (II) $op(L) = \cup \rightarrow sel(L) = \frac{1}{2}(1 - (1 - sel(L_1))(1 - sel(L_2)))$ and (3) $op(L) = \setminus \rightarrow sel(L) = \frac{1}{2}sel(L_1)(1 - sel(L_2))$. The results achieved with L_0 were our *Baseline*.

We first compared the execution time of C-RO and RO-MA (see Table 1) against the *Baseline* for all 7 datasets alongside with LIGER. As expected, all variations of LIGER require less execution time than the *Baseline*. As a result, LIGER produces more time-efficient LS, even when *maxOpt* is set to a high value. LIGER performs best on VILLAGES for $k = 0.1$ and $maxOpt = 0.4\,s$, where it can reduce the average runtime of the 100 LSs we considered by 88%. On the smaller BDLP-ACM dataset, RO-MA performs best and achieves a time reduction of the run time by 77.5%. Futhermore, we studied how the strategies C-RO and RO-MA compare to each other (see Table 1). Our average results suggest that RO-MA outperforms C-RO on average. The statistical significance of these results is confirmed by a paired t-test on the average run time

distributions (significance level = 0.95). Our intuition that the *monotonicity of run times* can potentially improve the run time of our approach is supported by the results on three out of the seven datasets (`Abt-Buy`, `DBLP-ACM` and `Amazon-GP`). On the remaining four datasets, RO-MA outperforms C-RO on average. Still, when C-RO outperforms RO-MA, the absolute differences are minute. Additionally, both subsumed LSs received the same selectivity. Hence, when the available refinement time is limited, RO-MA should be preferred when aiming to carry out partial-recall LD. The highest absolute difference between C-RO and RO-MA is achieved on the `DBLP-Scholar` dataset, where RO-MA is 1179.59 s faster than C-RO, while the highest relative gain of 776.28% by C-RO against the *Baseline* is achieved on `VILLAGES` (k=10%, $maxOpt = 400$), which is the largest dataset of our experiments.

Finally, we wanted to measure the loss of F-measure of a machine-learning approach when presented with the results of partial-recall LD vs. the F-measure it would achieve using the full results. We use WOMBAT [8], which is currently the only approach for learning LSs from positive examples. Our results show that with an expected partial recall of 50%, WOMBAT achieves at least 76.6% of the F-measure that it achieves when presented with all the data generated by EAGLE (recall = 100%).

5 Conclusions and Future Work

We presented LIGER, the first partial-recall LD approach. We provided a formal definition of a downward refinement operator along with its characteristics,which we used to develop an algorithm for partial-recall LD. We thus evaluated our approach on 7 datasets and showed that by using our refinement operator, we are able to detect LS with guaranteed expected recall efficiently. Our extension of the LIGER algorithm with a monotonicity assumption pertaining to the run time of the LS was shown to be slightly better than the basic LIGER implementation. In future work, we will build upon LIGER to guarantee the real selectivity and recall of our approaches with a given probability.

References

1. Georgala, K., Hoffmann, M., Ngomo, A.N.: An Evaluation of Models for Runtime Approximation in Link Discovery. In: Proceedings of the International Conference on WI (2017)
2. Georgala, K., Obraczka, D., Ngonga Ngomo, A.C.: Dynamic planning for link discovery. In: The Semantic Web. pp. 240–255 (2018)
3. Isele, R., Jentzsch, A., Bizer, C.: Efficient Multidimensional Blocking for Link Discovery without losing Recall. In: Marian, A., Vassalos, V. (eds.) WebDB (2011)
4. Köpcke, H., Thor, A., Rahm, E.: Evaluation of Entity Resolution Approaches on Real-world Match Problems. Proc. VLDB Endow. **3**(1-2), 484–493 (Sep 2010)
5. Ngomo, A.C.N., Lyko, K.: Eagle: Efficient active learning of link specifications using genetic programming. In: Extended Semantic Web Conference. pp. 149–163. Springer (2012)
6. Ngonga Ngomo, A.C.: On Link Discovery using a Hybrid Approach. Journal on Data Semantics **1**(4), 203–217 (2012), `http://dx.doi.org/10.1007/s13740-012-0012-y`
7. Ngonga Ngomo, A.C.: HELIOS – Execution Optimization for Link Discovery, pp. 17–32. Springer International Publishing, Cham (2014)
8. Sherif, M., Ngonga Ngomo, A.C., Lehmann, J.: WOMBAT - A Generalization Approach for Automatic Link Discovery. In: 14th Extended Semantic Web Conference. Springer (2017)

Results of the Ontology Alignment Evaluation Initiative 2020*

Mina Abd Nikooie Pour[1], Alsayed Algergawy[2], Reihaneh Amini[3], Daniel Faria[4], Irini Fundulaki[5], Ian Harrow[6], Sven Hertling[7], Ernesto Jiménez-Ruiz[8,9], Clement Jonquet[10], Naouel Karam[11], Abderrahmane Khiat[12], Amir Laadhar[10], Patrick Lambrix[1], Huanyu Li[1], Ying Li[1], Pascal Hitzler[3], Heiko Paulheim[7], Catia Pesquita[13], Tzanina Saveta[5], Pavel Shvaiko[14], Andrea Splendiani[6], Elodie Thiéblin[15], Cássia Trojahn[16], Jana Vataščinová[17], Beyza Yaman[18], Ondřej Zamazal[17], and Lu Zhou[3]

[1] Linköping University & Swedish e-Science Research Center, Linköping, Sweden
{mina.abd.nikooie.pour,patrick.lambrix,huanyu.li,ying.li}@liu.se
[2] Friedrich Schiller University Jena, Germany
alsayed.algergawy@uni-jena.de
[3] Data Semantics (DaSe) Laboratory, Kansas State University, USA
{luzhou,reihanea,hitzler}@ksu.edu
[4] BioData.pt, INESC-ID, Lisbon, Portugal
dfaria@inesc-id.pt
[5] Institute of Computer Science-FORTH, Heraklion, Greece
{jsaveta,fundul}@ics.forth.gr
[6] Pistoia Alliance Inc., USA
{ian.harrow,andrea.splendiani}@pistoiaalliance.org
[7] University of Mannheim, Germany
{sven,heiko}@informatik.uni-mannheim.de
[8] City, University of London, UK
ernesto.jimenez-ruiz@city.ac.uk
[9] Department of Informatics, University of Oslo, Norway
ernestoj@ifi.uio.no
[10] LIRMM, University of Montpellier & CNRS, France
{jonquet,amir.laadhar}@lirmm.fr
[11] Fraunhofer FOKUS, Berlin, Germany
naouel.karam@fokus.fraunhofer.de
[12] Fraunhofer IAIS, Sankt Augustin, Germany
abderrahmane.khiat@iais.fraunhofer.de
[13] LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
cpesquita@di.fc.ul.pt
[14] TasLab, Trentino Digitale SpA, Trento, Italy
pavel.shvaiko@tndigit.it
[15] Logilab, France
elodie.thieblin@logilab.fr
[16] IRIT & Université Toulouse II, Toulouse, France
cassia.trojahn@irit.fr
[17] University of Economics, Prague, Czech Republic
{jana.vatascinova,ondrej.zamazal}@vse.cz
[18] ADAPT Centre, Dublin City University, Ireland
beyza.yamanadaptcentre.ie

Abstract. The Ontology Alignment Evaluation Initiative (OAEI) aims at comparing ontology matching systems on precisely defined test cases. These test cases can be based on ontologies of different levels of complexity and use different evaluation modalities (e.g., blind evaluation, open evaluation, or consensus). The OAEI 2020 campaign offered 12 tracks with 36 test cases, and was attended by 19 participants. This paper is an overall presentation of that campaign.

1 Introduction

The Ontology Alignment Evaluation Initiative[1] (OAEI) is a coordinated international initiative, which organizes the evaluation of an increasing number of ontology matching systems [26, 28], and which has been run for seventeen years by now. The main goal of the OAEI is to compare systems and algorithms openly and on the same basis, in order to allow anyone to draw conclusions about the best matching strategies. Furthermore, the ambition is that, from such evaluations, developers can improve their systems and offer better tools that answer the evolving application needs.

Two first events were organized in 2004: *(i)* the Information Interpretation and Integration Conference (I3CON) held at the NIST Performance Metrics for Intelligent Systems (PerMIS) workshop and *(ii)* the Ontology Alignment Contest held at the Evaluation of Ontology-based Tools (EON) workshop of the annual International Semantic Web Conference (ISWC) [66]. Then, a unique OAEI campaign occurred in 2005 at the workshop on Integrating Ontologies held in conjunction with the International Conference on Knowledge Capture (K-Cap) [7]. From 2006 until the present, the OAEI campaigns were held at the Ontology Matching workshop, collocated with ISWC [5, 4, 1, 2, 11, 18, 15, 3, 24, 23, 22, 10, 25, 27], which this year took place virtually (originally planned in Athens, Greece)[2].

Since 2011, we have been using an environment for automatically processing evaluations (Section 2.1) which was developed within the SEALS (Semantic Evaluation At Large Scale) project[3]. SEALS provided a software infrastructure for automatically executing evaluations and evaluation campaigns for typical semantic web tools, including ontology matching. Since OAEI 2017, a novel evaluation environment, called HOBBIT (Section 2.1), was adopted for the HOBBIT Link Discovery track, and later extended to enable the evaluation of other tracks. Some tracks are run exclusively through SEALS and others through HOBBIT, but several allow participants to choose the platform they prefer. This year, the MELT framework [36] was adopted in order to facilitate the SEALS and HOBBIT wrapping and evaluation.

This paper synthesizes the 2020 evaluation campaign and introduces the results provided in the papers of the participants. The remainder of the paper is organized as follows: in Section 2, we present the overall evaluation methodology; in Section 3 we present the tracks and datasets; in Section 4 we present and discuss the results; and finally, Section 5 discusses the lessons learned.

[1] http://oaei.ontologymatching.org

[2] http://om2020.ontologymatching.org

[3] http://www.seals-project.eu

2 Methodology

2.1 Evaluation platforms

The OAEI evaluation was carried out in one of two alternative platforms: the SEALS client or the HOBBIT platform. Both have the goal of ensuring reproducibility and comparability of the results across matching systems.

The **SEALS client** was developed in 2011. It is a Java-based command line interface for ontology matching evaluation, which requires system developers to implement a simple interface and to wrap their tools in a predefined way including all required libraries and resources. A tutorial for tool wrapping is provided to the participants, describing how to wrap a tool and how to run a full evaluation locally.

The **HOBBIT platform**[4] was introduced in 2017. It is a web interface for linked data and ontology matching evaluation, which requires systems to be wrapped inside docker containers and includes a SystemAdapter class, then being uploaded into the HOBBIT platform [44].

Both platforms compute the standard evaluation metrics against the reference alignments: precision, recall and F-measure. In test cases where different evaluation modalities are required, evaluation was carried out *a posteriori*, using the alignments produced by the matching systems.

The **MELT framework**[5] [36] was introduced in 2019 and is under active development. It allows to develop, evaluate, and package matching systems for arbitrary evaluation interfaces like SEALS or HOBBIT. It further enables developers to use Python in their matching systems. In terms of evaluation, MELT offers a correspondence level analysis for multiple matching systems which can even implement different interfaces. It is, therefore, suitable for track organisers as well as system developers.

2.2 OAEI campaign phases

As in previous years, the OAEI 2020 campaign was divided into three phases: preparatory, execution, and evaluation.

In the **preparatory phase**, the test cases were provided to participants in an initial assessment period between June 15^{th} and July 15^{th}, 2020. The goal of this phase is to ensure that the test cases make sense to participants, and give them the opportunity to provide feedback to organizers on the test case as well as potentially report errors. At the end of this phase, the final test base was frozen and released.

During the ensuing **execution phase**, participants test and potentially develop their matching systems to automatically match the test cases. Participants can self-evaluate their results either by comparing their output with the reference alignments or by using either of the evaluation platforms. They can tune their systems with respect to the non-blind evaluation as long as they respect the rules of the OAEI. Participants were required to register their systems and make a preliminary evaluation by July 31^{st}. The execution phase was terminated on October 15^{th}, 2020, at which date participants had to submit the (near) final versions of their systems (SEALS-wrapped and/or HOBBIT-wrapped).

[4] `https://project-hobbit.eu/outcomes/hobbit-platform/`
[5] `https://github.com/dwslab/melt`

During the **evaluation phase**, systems were evaluated by all track organizers. In case minor problems were found during the initial stages of this phase, they were reported to the developers, who were given the opportunity to fix and resubmit their systems. Initial results were provided directly to the participants, whereas final results for most tracks were published on the respective OAEI web pages by October 24^{th}, 2020.

3 Tracks and test cases

This year's OAEI campaign consisted of 12 tracks gathering 36 test cases, all of which included OWL ontologies to align.[6] They can be grouped into:

- Schema matching tracks, which have as objective matching ontology classes and/or properties.
- Instance Matching tracks, which have as objective matching ontology instances.
- Instance and Schema Matching tracks, which involve both of the above.
- Complex Matching tracks, which have as objective finding complex correspondences between ontology entities.
- Interactive tracks, which simulate user interaction to enable the benchmarking of interactive matching algorithms.

The tracks are summarized in Table 1.

3.1 Anatomy

The anatomy track comprises a single test case consisting of matching two fragments of biomedical ontologies which describe the human anatomy[7] (3304 classes) and the anatomy of the mouse[8] (2744 classes). The evaluation is based on a manually curated reference alignment. This dataset has been used since 2007 with some improvements over the years [20].

Systems are evaluated with the standard parameters of precision, recall, F-measure. Additionally, recall+ is computed by excluding trivial correspondences (i.e., correspondences that have the same normalized label). Alignments are also checked for coherence using the Pellet reasoner. The evaluation was carried out on a server with a 6 core CPU @ 3.46 GHz with 8GB allocated RAM, using the SEALS client. For some system requires more RAM, the evaluation was carried out on a Windows 10 (64-bit) desktop with an Intel Core i7-6700 CPU @ 3.40GHz x 8 with 16GB RAM allocated. However, the evaluation parameters were computed *a posteriori*, after removing from the alignments produced by the systems, correspondences expressing relations other than equivalence, as well as trivial correspondences in the oboInOwl namespace (e.g., oboInOwl#Synonym = oboInOwl#Synonym). The results obtained with the SEALS client vary in some cases by 0.5% compared to the results presented below.

[6] The Biodiversity and Ecology track also included SKOS thesauri.
[7] `www.cancer.gov/cancertopics/cancerlibrary/terminologyresources`
[8] `http://www.informatics.jax.org/searches/AMA_form.shtml`

Table 1. Characteristics of the OAEI tracks.

Track	Test Cases (Tasks)	Relations	Confidence	Evaluation	Languages	Platform
Schema Matching						
Anatomy	1	=	[0 1]	open	EN	SEALS
Biodiversity & Ecology	4	=	[0 1]	open	EN	SEALS
Conference	1 (21)	=, <=	[0 1]	open+blind	EN	SEALS
Disease & Phenotype	2	=, <=	[0 1]	open+blind	EN	SEALS
Large Biomedical ontologies	6	=	[0 1]	open	EN	both
Multifarm	2 (2445)	=	[0 1]	open+blind	AR, CZ, CN, DE, EN, ES, FR, IT, NL, RU, PT	SEALS
Instance Matching						
Link Discovery	2 (9)	=	[0 1]	open	EN	HOBBIT
SPIMBENCH	2	=	[0 1]	open+blind	EN	HOBBIT
Geolink Cruise	4	=	[0 1]	open	EN	SEALS
Instance and Schema Matching						
Knowledge Graph	5	=	[0 1]	open+blind	EN	SEALS
Interactive Matching						
Interactive	2 (22)	=, <=	[0 1]	open	EN	SEALS
Complex Matching						
Complex	7	=, <=, >=	[0 1]	open+blind	EN, ES	SEALS

Open evaluation is made with already published reference alignments and blind evaluation is made by organizers, either from reference alignments unknown to the participants or manually.

3.2 Biodiversity and Ecology

The biodiversity and ecology (biodiv) track has been originally motivated by two projects, namely GFBio[9] (The German Federation for Biological Data) and AquaDiva[10], which aim at providing semantically enriched data management solutions for data capture, annotation, indexing and search [46, 48]. This year, the third edition of the biodiv track features the two matching tasks present in former editions, namely: matching the Environment Ontology (ENVO) [9] to the Semantic Web for Earth and Environment Technology Ontology (SWEET) [58], and matching the Flora Phenotype Ontology (FLOPO) [38] to Plant Trait Ontology (PTO) [14]. In this edition, we partnered with the D2KAB project[11] (Data to Knowledge in Agronomy and Biodiversity) which develops the AgroPortal[12] vocabulary and ontology repository, to include

[9] www.gfbio.org
[10] www.aquadiva.uni-jena.de
[11] www.d2kab.org
[12] agroportal.lirmm.fr

two new matching tasks involving important thesauri (originally developed in SKOS) in agronomy and environmental sciences: finding alignments between the AGROVOC thesaurus [59] and the US National Agricultural Library Thesaurus (NALT)[13] and between the General Multilingual Environmental Thesaurus (GEMET)[14] and the Analysis and Experimentation on Ecosystems thesaurus (ANAEETHES)[13]. These ontologies and thesauri are particularly useful for biodiversity and ecology research and are being used in various projects. They have been developed in parallel and are significantly overlapping. They are semantically rich and contain tens of thousands of concepts. By providing semantic resources developed in SKOS, our objective is also to encourage the ontology alignment community to develop tools that can natively handle SKOS which is an important standard to encode terminologies (particularly thesauri and taxonomies) and for which alignment is also very important.

Table 2 presents detailed information about the ontologies and thesauri used in the evaluation, such as the ontology format, version, number of classes as well as the number of instances[15].

Table 2. Version, format and number of classes of the Biodiversity and Ecology track ontologies and thesauri.

Ontology/Thesaurus	Format	Version	Classes	Instances
ENVO	OWL	2020-03-08	9053	-
SWEET	OWL	2019-10-12	4533	-
FLOPO	OWL	2016-06-03	28965	-
PTO	OWL	2017-09-11	1504	-
AGROVOC	SKOS	2020-10-02	46	706803
NALT	SKOS	2020-28-01	2	74158
GEMET	SKOS	2020-13-02	7	5907
ANAEETHES	SKOS	2017-22-03	2	3323

For the ontologies ENVO, SWEET, FLOPO and PTO, we created the reference alignments for the tasks following the same procedure as in former editions. Reference files were produced using a hybrid approach consisting of (1) a consensus alignment based on matching systems output, then (2) manually validating a subset of unique mappings produced by each system (and adding them to the consensus if considered correct), and finally (3) adding a set of manually generated correspondences. The matching systems used to generate the consensus alignments were those participating to this track in 2018 [4], namely: AML, Lily, the LogMap family, POMAP and XMAP.

[13] `agclass.nal.usda.gov`

[14] `www.eionet.europa.eu/gemet`

[15] Note that SKOS thesauri conceptualize by means of instances of `skos:Concept` and not `owl:Class`. Still, the *biodiv* track is different from instance matching tracks, as in both cases concepts or classes are used to define the structure (or schema) of a semantic resource.

For the thesauri AGROVOC, NALT, GEMET and ANEETHES, we created the reference alignments using the Ontology Mapping Harvesting Tool (OMHT).[16] OMHT was developed as a standalone Java program that works with one semantic resource file pulled out from AgroPortal or BioPortal[17]. OMHT automatically extracts all declared mappings by developers inside an ontology or a thesauri source files. We used for the reference alignments only the mappings with a `skos:exactMatch` property.

The evaluation was carried out on a Windows 10 (64-bit) desktop with an Intel Core i7-4770 CPU @ 3.40GHz x 4 with 16 GB RAM allocated, using the SEALS client. Systems were evaluated using the standard metrics.

3.3 Conference

The conference track features a single test case that is a suite of 21 matching tasks corresponding to the pairwise combination of 7 moderately expressive ontologies describing the domain of organizing conferences. The dataset and its usage are described in [70].

The track uses several reference alignments for evaluation: the old (and not fully complete) manually curated open reference alignment, *ra1*; an extended, also manually curated version of this alignment, *ra2*; a version of the latter corrected to resolve violations of conservativity, *rar2*; and an uncertain version of *ra1* produced through crowd-sourcing, where the score of each correspondence is the fraction of people in the evaluation group that agree with the correspondence. The latter reference was used in two evaluation modalities: *discrete* and *continuous* evaluation. In the former, correspondences in the uncertain reference alignment with a score of at least 0.5 are treated as correct whereas those with lower score are treated as incorrect, and standard evaluation parameters are used to evaluated systems. In the latter, weighted precision, recall and F-measure values are computed by taking into consideration the actual scores of the uncertain reference, as well as the scores generated by the matching system. For the sharp reference alignments (*ra1*, *ra2* and *rar2*), the evaluation is based on the standard parameters, as well the $F_{0.5}$-measure and F_2-measure and on conservativity and consistency violations. Whereas F_1 is the harmonic mean of precision and recall where both receive equal weight, F_2 gives higher weight to recall than precision and $F_{0.5}$ gives higher weight to precision higher than recall. The track also includes an analysis of False Positives.

Two baseline matchers are used to benchmark the systems: edna string edit distance matcher; and StringEquiv string equivalence matcher as in the anatomy test case.

The evaluation was carried out on a Windows 10 (64-bit) desktop with an Intel Core i7–8550U (1,8 GHz, TB 4 GHz) x 4 with 16 GB RAM allocated using the SEALS client. Systems were evaluated using the standard metrics.

3.4 Disease and Phenotype

The Disease and Phenotype is organized by the Pistoia Alliance Ontologies Mapping project team[18]. It comprises 2 test cases that involve 4 biomedical ontologies cov-

[16] `https://github.com/agroportal/ontology_mapping_harvester`
[17] `https://bioportal.bioontology.org`
[18] `http://www.pistoiaalliance.org/projects/ontologies-mapping/`

ering the disease and phenotype domains: Human Phenotype Ontology (HP) versus Mammalian Phenotype Ontology (MP) and Human Disease Ontology (DOID) versus Orphanet and Rare Diseases Ontology (ORDO). Currently, correspondences between these ontologies are mostly curated by bioinformatics and disease experts who would benefit from automation of their workflows supported by implementation of ontology matching algorithms. More details about the Pistoia Alliance Ontologies Mapping project and the OAEI evaluation are available in [31]. Table 3 summarizes the versions of the ontologies used in OAEI 2020.

Table 3. Disease and Phenotype ontology versions and sources.

Ontology	Version	Source
HP	2017-06-30	OBO Foundry
MP	2017-06-29	OBO Foundry
DOID	2017-06-13	OBO Foundry
ORDO	v2.4	ORPHADATA

The reference alignments used in this track are silver standard consensus alignments automatically built by merging/voting the outputs of the participating systems in the OAEI campaigns 2016-2020 (with vote=3). Note that systems participating with different variants and in different years only contributed once in the voting, that is, the voting was done by family of systems/variants rather than by individual systems. The HP-MP silver standard thus contains 2,504 correspondences, whereas the DOID-ORDO one contains 3,909 correspondences.

Systems were evaluated using the standard parameters as well as the (approximate) number of unsatisfiable classes computed using the OWL 2 EL reasoner ELK [47]. The evaluation was carried out in a Ubuntu 18 Laptop with an Intel Core i5-6300HQ CPU @ 2.30GHz x 4 and allocating 15 Gb of RAM.

3.5 Large Biomedical Ontologies

The large biomedical ontologies (largebio) track aims at finding alignments between the large and semantically rich biomedical ontologies FMA, SNOMED-CT, and NCI, which contain 78,989, 306,591 and 66,724 classes, respectively. The track consists of six test cases corresponding to three matching problems (FMA-NCI, FMA-SNOMED and SNOMED-NCI) in two modalities: small overlapping fragments and whole ontologies (FMA and NCI) or large fragments (SNOMED-CT).

The reference alignments used in this track are derived directly from the UMLS Metathesaurus [8] as detailed in [42], then automatically repaired to ensure logical coherence. However, rather than use a standard repair procedure of removing problem causing correspondences, we set the relation of such correspondences to *"?"* (unknown). These "?" correspondences are neither considered positive nor negative when evaluating matching systems, but are simply ignored. This way, systems that do not perform alignment repair are not penalized for finding correspondences that (despite causing incoherences) may or may not be correct, and systems that do perform alignment

repair are not penalized for removing such correspondences. To avoid any bias, correspondences were considered problem causing if they were selected for removal by any of the three established repair algorithms: Alcomo [52], LogMap [41], or AML [60]. The reference alignments are summarized in Table 4.

Table 4. Number of correspondences in the reference alignments of the large biomedical ontologies tasks.

Reference alignment	"=" corresp.	"?" corresp.
FMA-NCI	2,686	338
FMA-SNOMED	6,026	2,982
SNOMED-NCI	17,210	1,634

The evaluation was carried out in a Ubuntu 18 Laptop with an Intel Core i5-6300HQ CPU @ 2.30GHz x 4 and allocating 15 Gb of RAM. Evaluation was based on the standard parameters (modified to account for the "?" relations) as well as the number of unsatisfiable classes and the ratio of unsatisfiable classes with respect to the size of the union of the input ontologies. Unsatisfiable classes were computed using the OWL 2 reasoner HermiT [54], or, in the cases in which HermiT could not cope with the input ontologies and the alignments (in less than 2 hours) a lower bound on the number of unsatisfiable classes (indicated by $\geq$) was computed using the OWL2 EL reasoner ELK [47].

3.6 Multifarm

The multifarm track [53] aims at evaluating the ability of matching systems to deal with ontologies in different natural languages. This dataset results from the translation of 7 ontologies from the conference track (cmt, conference, confOf, iasted, sigkdd, ekaw and edas) into 10 languages: Arabic (ar), Chinese (cn), Czech (cz), Dutch (nl), French (fr), German (de), Italian (it), Portuguese (pt), Russian (ru), and Spanish (es). The dataset is composed of 55 pairs of languages, with 49 matching tasks for each of them, taking into account the alignment direction (e.g. $\text{cmt}_{en} \rightarrow \text{edas}_{de}$ and $\text{cmt}_{de} \rightarrow \text{edas}_{en}$ are distinct matching tasks). While part of the dataset is openly available, all matching tasks involving the *edas* and *ekaw* ontologies (resulting in 55×24 matching tasks) are used for blind evaluation.

We consider two test cases: i) those tasks where two different ontologies (cmt→edas, for instance) have been translated into two different languages; and ii) those tasks where the same ontology (cmt→cmt) has been translated into two different languages. For the tasks of type ii), good results are not only related to the use of specific techniques for dealing with cross-lingual ontologies, but also on the ability to exploit the identical structure of the ontologies.

The reference alignments used in this track derive directly from the manually curated Conference *ra1* reference alignments. The systems have been executed on a Ubuntu Linux machine configured with 8GB of RAM running under a Intel Core CPU 2.00GHz x4 processors, using the SEALS client.

3.7 Link Discovery

The Link Discovery track features two test cases, Linking and Spatial, that deal with *link discovery* for spatial data represented as *trajectories* i.e., sequences of longitude, latitude pairs. The track is based on two datasets generated from TomTom[19] and Spaten [17].

The **Linking** test case aims at testing the performance of instance matching tools that implement mostly string-based approaches for identifying matching entities. It can be used not only by instance matching tools, but also by SPARQL engines that deal with query answering over geospatial data. The test case was based on SPIMBENCH [62], but since the ontologies used to represent trajectories are fairly simple and do not consider complex RDF or OWL schema constructs already supported by SPIMBENCH, only a subset of the transformations implemented by SPIMBENCH was used. The transformations implemented in the test case were (i) string-based with different (a) levels, (b) types of spatial object representations and (c) types of date representations, and (ii) schema-based, i.e., addition and deletion of ontology (schema) properties. These transformations were implemented in the TomTom dataset. In a nutshell, instance matching systems are expected to determine whether two traces with their points annotated with place names designate the same trajectory. In order to evaluate the systems a ground truth was built that contains the set of expected links where an instance s_1 in the source dataset is associated with an instance t_1 in the target dataset that has been generated as a modified description of s_1.

The **Spatial** test case aims at testing the performance of systems that deal with topological relations proposed in the state of the art DE-9IM (Dimensionally Extended nine-Intersection Model) model [65]. The benchmark generator behind this test case implements all topological relations of DE-9IM between trajectories in the two dimensional space. To the best of our knowledge such a generic benchmark, that takes as input trajectories and checks the performance of linking systems for spatial data does not exist. The focus for the design was (a) on the correct implementation of all the topological relations of the DE-9IM topological model and (b) on producing datasets large enough to stress the systems under test. The supported relations are: *Equals, Disjoint, Touches, Contains/Within, Covers/CoveredBy, Intersects, Crosses, Overlaps*. The test case comprises tasks for all the DE-9IM relations and for LineString/LineString and LineString/Polygon cases, for both TomTom and Spaten datasets, ranging from 200 to 2K instances. We did not exceed 64 KB per instance due to a limitation of the Silk system[20], in order to enable a fair comparison of the systems participating in this track.

The evaluation for both test cases was carried out using the HOBBIT platform.

3.8 SPIMBENCH

The **SPIMBENCH** track consists of matching instances that are found to refer to the same real-world entity corresponding to a creative work (that can be a news item,

[19] `https://www.tomtom.com/en_gr/`

[20] `https://github.com/silk-framework/silk/issues/57`

blog post or programme). The datasets were generated and transformed using SPIMBENCH [62] by altering a set of original linked data through value-based, structure-based, and semantics-aware transformations (simple combination of transformations). They share almost the same ontology (with some differences in property level, due to the structure-based transformations), which describes instances using 22 classes, 31 data properties, and 85 object properties. Participants are requested to produce a set of correspondences between the pairs of matching instances from the source and target datasets that are found to refer to the same real-world entity. An instance in the source dataset can have none or one matching counterpart in the target dataset. The SPIMBENCH task uses two sets of datasets[21] with different scales (i.e., number of instances to match):

– Sandbox (380 INSTANCES, 10000 TRIPLES). It contains two datasets called source (Tbox1) and target (Tbox2) as well as the set of expected correspondences (i.e., reference alignment).
– Mainbox (1800 CWs, 50000 TRIPLES). It contains two datasets called source (Tbox1) and target (Tbox2). This test case is blind, meaning that the reference alignment is not given to the participants.

In both cases, the goal is to discover the correspondences among the instances in the source dataset (Tbox1) and the instances in the target dataset (Tbox2).

The evaluation was carried out using the HOBBIT platform.

3.9 Geolink Cruise

The **Geolink Cruise** track consists of matching instances from different ontologies describing the same cruise in the real-world. The datasets are collected from the Geolink project,[22] which was funded under the U.S. National Science Foundation's EarthCube initiative. The datasets and alignments are guaranteed to contain real-world use cases to solve the instance matching problem in practice. In the GeoLink Cruise dataset, there are two ontologies which are GeoLink Base Ontology (gbo) and GeoLink Modular Ontology (gmo). The data providers from different organizations populate their own data into these two ontologies. In this track, we utilize instances from two different data providers, Biological and Chemical Oceanography Data Management Office (bco-

[21] Although the files are called Tbox1 and Tbox2, they actually contain a Tbox and an Abox.
[22] `https://www.geolink.org/`

Table 5. The Statistics of the Ontologies in the Geolink Cruise.

Ontology	#Class	#Object Property	#Data Property	#Individual	#Triple
gbo_bco-dmo	40	149	49	1061	13055
gbo_r2r	40	149	49	5320	27992
gmo_bco-dmo	79	79	37	1052	16303
gmo_r2r	79	79	37	2025	24798

dmo)[23] and Rolling Deck to Repository (r2r)[24] and populate all the triples related to Cruise into two ontologies. There are 491 Cruise pairs between these two datasets that are labelled by domain experts as equivalent. Some statistic information of the ontologies are listed in the Table 5. More details of this benchmark can be found in the paper [6].

3.10 Knowledge Graph

The Knowledge Graph track was run for the third year. The task of the track is to match pairs of knowledge graphs, whose schema and instances have to be matched simultaneously. The individual knowledge graphs are created by running the DBpedia extraction framework on eight different Wikis from the Fandom Wiki hosting platform[25] in the course of the DBkWik project [34, 33]. They cover different topics (movies, games, comics and books) and three Knowledge Graph clusters sharing the same domain e.g. star trek, as shown in Table 6.

Table 6. Characteristics of the Knowledge Graphs in the Knowledge Graph track, and the sources they were created from.

Source	Hub	Topic	#Instances	#Properties	#Classes
Star Wars Wiki	Movies	Entertainment	145,033	700	269
The Old Republic Wiki	Games	Gaming	4,180	368	101
Star Wars Galaxies Wiki	Games	Gaming	9,634	148	67
Marvel Database	Comics	Comics	210,996	139	186
Marvel Cinematic Universe	Movies	Entertainment	17,187	147	55
Memory Alpha	TV	Entertainment	45,828	325	181
Star Trek Expanded Universe	TV	Entertainment	13,426	202	283
Memory Beta	Books	Entertainment	51,323	423	240

The evaluation is based on reference correspondences at both schema and instance levels. While the schema level correspondences were created by experts, the instance correspondences were extracted from the wiki page itself. Due to the fact that not all inter wiki links on a page represent the same concept a few restrictions were made: 1) only links in sections with a header containing "link" are used, 2) all links are removed where the source page links to more than one concept in another wiki (ensures the alignments are functional), 3) multiple links which point to the same concept are also removed (ensures injectivity), 4) links to disambiguation pages were manually checked and corrected. Since we do not have a correspondence for each instance, class, and property in the graphs, this gold standard is only a *partial gold standard*.

The evaluation was executed on a virtual machine (VM) with 32GB of RAM and 16 vCPUs (2.4 GHz), with Debian 9 operating system and Openjdk version 1.8.0_265,

[23] `https://www.bco-dmo.org/`
[24] `https://www.rvdata.us/`
[25] `https://www.wikia.com/`

using the SEALS client (version 7.0.5). The -o option in SEALS is used to provide the two knowledge graphs which should be matched. This decreases runtime because the matching system can load the input from local files rather than downloading it from HTTP URLs. We could not use the "-x" option of SEALS because the evaluation routine needed to be changed for two reasons: first, to differentiate between results for class, property, and instance correspondences, and second, to deal with the partial nature of the gold standard.

The alignments were evaluated based on precision, recall, and f-measure for classes, properties, and instances (each in isolation). The partial gold standard contained 1:1 correspondences and we further assume that in each knowledge graph, only one representation of the concept exists. This means that if we have a correspondence in our gold standard, we count a correspondence to a different concept as a false positive. The count of false negatives is only increased if we have a 1:1 correspondence and it is not found by a matcher. The whole source code for generating the evaluation results is also available.[26]

Additionally we run the matchers on three hidden test cases where the source wikis are: Marvel Cinematic Universe, Memory Alpha, and Star Wars Wiki. The target wiki is for all test cases the same. It is the lyrics wiki with 1,062,920 instances, 270 properties and 67 classes. The goal is to explore how the matchers behave on matching mostly *unrelated* knowledge graphs.

As a baseline, we employed two simple string matching approaches. The source code for these matchers is publicly available.[27]

3.11 Interactive Matching

The interactive matching track aims to assess the performance of semi-automated matching systems by simulating user interaction [56, 19, 50]. The evaluation thus focuses on how interaction with the user improves the matching results. Currently, this track does not evaluate the user experience or the user interfaces of the systems [39, 19].

The interactive matching track is based on the datasets from the Anatomy and Conference tracks, which have been previously described. It relies on the SEALS client's *Oracle* class to simulate user interactions. An interactive matching system can present a collection of correspondences simultaneously to the oracle, which will tell the system whether that correspondence is correct or not. If a system presents up to three correspondences together and each correspondence presented has a mapped entity (i.e., class or property) in common with at least one other correspondence presented, the oracle counts this as a single interaction, under the rationale that this corresponds to a scenario where a user is asked to choose between conflicting candidate correspondences. To simulate the possibility of user errors, the oracle can be set to reply with a given error probability (randomly, from a uniform distribution). We evaluated systems with four different error rates: 0.0 (perfect user), 0.1, 0.2, and 0.3.

[26] http://oaei.ontologymatching.org/2020/results/knowledgegraph/matching-eval-trackspecific.zip

[27] http://oaei.ontologymatching.org/2019/results/knowledgegraph/kgBaselineMatchers.zip

In addition to the standard evaluation parameters, we also compute the number of requests made by the system, the total number of distinct correspondences asked, the number of positive and negative answers from the oracle, the performance of the system according to the oracle (to assess the impact of the oracle errors on the system) and finally, the performance of the oracle itself (to assess how erroneous it was).

The evaluation was carried out on a server with 3.46 GHz (6 cores) and 8GB RAM allocated to the matching systems. For systems requiring more RAM, the evaluation was carried out on a Windows 10 (64-bit) desktop with an Intel Core i7-6700 CPU @ 3.40GHz x 8 with 16GB RAM allocated. Each system was run ten times and the final result of a system for each error rate represents the average of these runs. For the Conference dataset with the ra1 alignment, precision and recall correspond to the micro-average over all ontology pairs, whereas the number of interactions is the total number of interactions for all the pairs.

3.12 Complex Matching

The complex matching track is meant to evaluate the matchers based on their ability to generate complex alignments. A complex alignment is composed of complex correspondences typically involving more than two ontology entities, such as o_1*:AcceptedPaper* $\equiv$ o_2*:Paper* $\sqcap$ o_2*:hasDecision.*o_2*:Acceptance*. In addition to last year's datasets [69], two new datasets have been added: Populated Geolink and Populated Enslaved.

The **complex conference** dataset is composed of three ontologies: cmt, conference and ekaw from the conference dataset. The reference alignment was created as a consensus between experts. In the evaluation process, the matchers can take the simple reference alignment *ra1* as input. The precision and recall measures are manually calculated over the complex equivalence correspondences only.

The **populated complex conference** is a populated version of the Conference dataset. 5 ontologies have been populated with more or less common instances resulting in 6 datasets (6 versions on the seals repository: v0, v20, v40, v60, v80 and v100). The alignments were evaluated based on Competency Questions for Alignment, i.e., basic queries that the alignment should be able to cover [67]. The queries are automatically rewritten using 2 systems: that from [68] which covers (1:n) correspondences with EDOAL expressions; and a system which compares the answers (sets of instances or sets of pairs of instances) of the source query and the source member of the correspondences and which outputs the target member if both sets are identical. The best rewritten query scores are kept. A precision score is given by comparing the instances described by the source and target members of the correspondences.

The **Hydrography** dataset consists of matching four different source ontologies (hydro3, hydrOntology-translated, hydrOntology-native, and cree) to a single target ontology (SWO) [12]. The evaluation process is based on three subtasks: given an entity from the source ontology, identify all related entities in the source and target ontology; given an entity in the source ontology and the set of related entities, identify the logical relation that holds between them; identify the full complex correspondences. The three subtasks were evaluated based on relaxed precision and recall [21].

The **GeoLink** dataset derives from the homonymous project, funded under the U.S. National Science Foundation's EarthCube initiative. It is composed of two ontologies: the GeoLink Base Ontology (GBO) and the GeoLink Modular Ontology (GMO). The GeoLink project is a real-world use case of ontologies. The alignment between the two ontologies was developed in consultation with domain experts from several geoscience research institutions. More detailed information on this benchmark can be found in [72]. Evaluation was done in the same way as with the Hydrography dataset. The evaluation platform was a MacBook Pro with a 2.5 GHz Intel Core i7 processor and 16 GB of 1600 MHz DDR3 RAM running mac OS Catalina version 10.15.6.

The **Populated GeoLink** dataset is designed to allow alignment systems that rely on the instance data to participate over the Geolink benchmark. The instance data are from real-worlds and collected from seven data repositories in the Geolink project. More detailed information on this benchmark can be found in [73]. Evaluation was done in the same way as with the Hydrography dataset. The evaluation platform was a MacBook Pro with a 2.5 GHz Intel Core i7 processor and 16 GB of 1600 MHz DDR3 RAM running mac OS Catalina version 10.15.6.

The **Populated Enslaved** dataset was derived from the ongoing project entitled "Enslaved: People of the Historical Slave Trade[28] and funded by The Andrew W. Mellon Foundation where the focus is on tracking the movements and details of peoples in the historical slave trade. It is composed of the Enslaved ontology and the Enslaved Wikibase repository along with the populated instance data. To the best of our knowledge, it is the first attempt to align a modular ontology to the Wikibase repository. More detailed information on this benchmark can be found in [71]. Evaluation was done in the same way as with the Hydrography dataset. The evaluation platform was a MacBook Pro with a 2.5 GHz Intel Core i7 processor and 16 GB of 1600 MHz DDR3 RAM running mac OS Catalina version 10.15.6.

The **Taxon** dataset is composed of four knowledge bases containing knowledge about plant taxonomy: AgronomicTaxon, AGROVOC, TAXREF-LD and DBpedia. The evaluation is two-fold: first, the precision of the output alignment is manually assessed; then, a set of source queries are rewritten using the output alignment. The rewritten target query is then manually classified as correct or incorrect. A source query is considered successfully rewritten if at least one of the target queries is semantically equivalent to it. The proportion of source queries successfully rewritten is then calculated (QWR in the results table). The evaluation over this dataset is open to all matching systems (simple or complex) but some queries can not be rewritten without complex correspondences. The evaluation was performed with an Ubuntu 16.04 machine configured with 16GB of RAM running under a i7-4790K CPU 4.00GHz x 8 processors.

4 Results and Discussion

4.1 Participation

Following an initial period of growth, the number of OAEI participants has remained approximately constant since 2012, which is slightly over 20. This year we count with

[28] `https://enslaved.org/`

19 participating systems. Table 7 lists the participants and the tracks in which they competed. Some matching systems participated with different variants (AML, LogMap) whereas others were evaluated with different configurations, as requested by developers (see test case sections for details).

Table 7. Participants and the status of their submissions.

System	ALIN	ALOD2Vec	AML	AMLC	AROA	ATBox	DESKMatcher	CANARD	FTRLIM	Lily	LogMap	LogMap-Bio	LogMapLt	OntoConnect	RADON	RE-miner	Silk	VeeAlign	WktMtchr	Total=19
Confidence		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓		✓	✓	✓	16
anatomy	●	●	●	○	○	●	●	○	○	●	●	●	●	●	○	○	○	○	●	11
conference	●	●	●	○	○	●	●	○	○	●	●	○	●	○	○	○	○	●	●	10
multifarm	○	○	●	○	○	○	○	○	○	◐	●	○	●	○	○	○	○	●	●	6
complex	○	○	○	●	●	○	○	●	○	○	○	○	○	○	○	○	○	○	○	3
interactive	●	○	●	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	3
largebio	○	◐	●	○	○	●	◐	○	○	○	●	●	●	○	○	○	○	○	●	8
phenotype	○	●	●	○	○	●	○	○	○	○	●	●	●	○	○	○	○	○	●	7
biodiv	○	◐	●	○	○	●	○	○	○	○	●	●	●	○	○	○	○	○	◐	7
spimbench	○	○	●	○	○	○	○	○	●	●	●	○	○	○	○	●	○	○	○	5
link discovery	○	○	●	○	○	○	○	○	○	○	○	○	○	○	●	○	●	○	○	3
geolink cruise	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	0
knowledge graph	○	●	●	○	○	●	●	○	○	○	●	●	◐	○	○	○	○	○	●	8
total	3	6	10	1	1	6	4	1	1	4	9	5	7	1	1	1	1	2	7	71

Confidence pertains to the confidence scores returned by the system, with ✓ indicating that they are non-boolean; ○ indicates that the system did not participate in the track; ● indicates that it participated fully in the track; and ◐ indicates that it participated in or completed only part of the tasks of the track.

A number of participating systems use external sources of background knowledge, which are especially critical in matching ontologies in the biomedical domain. LogMap-Bio uses BioPortal as mediating ontology provider, that is, it retrieves from BioPortal the most suitable top-10 ontologies for each matching task. LogMap uses normalizations and spelling variants from the general (biomedical) purpose SPECIALIST Lexicon. AML has three sources of background knowledge which can be used as mediators between the input ontologies: the Uber Anatomy Ontology (Uberon), the Human Disease Ontology (DOID) and the Medical Subject Headings (MeSH). XMAP and Lily use a dictionary of synonyms (pre)extracted from the UMLS Metathesaurus. In addition Lily also uses a dictionary of synonyms (pre)extracted from BioPortal.

4.2 Anatomy

The results for the Anatomy track are shown in Table 8. Of the 11 systems participating

Table 8. Anatomy results, ordered by F-measure. Runtime is measured in seconds; "size" is the number of correspondences in the generated alignment.

System	Runtime	Size	Precision	F-measure	Recall	Recall+	Coherent
AML	29	1471	0.956	0.941	0.927	0.81	√
Lily	706	1517	0.901	0.901	0.902	0.747	-
LogMapBio	1005	1544	0.885	0.893	0.902	0.74	√
LogMap	7	1397	0.918	0.88	0.846	0.593	√
Wiktionary	65	1194	0.956	0.842	0.753	0.346	-
ALIN	1182	1107	0.986	0.832	0.72	0.382	√
LogMapLite	2	1147	0.962	0.828	0.728	0.288	-
ATBox	192	1030	0.987	0.799	0.671	0.129	-
ALOD2Vec	236	1403	0.83	0.798	0.768	0.386	-
OntoConnect	248	1012	0.996	0.797	0.665	0.136	-
StringEquiv	-	946	0.997	0.766	0.622	0.000	-
DESKMatcher	391	2002	0.472	0.537	0.623	0.023	-

in the Anatomy track, 10 achieved an F-measure higher than the StringEquiv baseline. Three systems were first time participants (ATBox, OntoConnect, and DESKMatcher). Long-term participating systems showed few changes in comparison with previous years with respect to alignment quality (precision, recall, F-measure, and recall+), size and run time. The exceptions were ALIN which increased in precision (from 0.974 to 0.986), recall (from 0.698 to 0.72), recall+ (from 0.365 to 0.382), F-measure (from 0.813 to 0.832), and size (from 1086 to 1107), and Lily that increased in precision (from 0.873 to 0.901), recall (from 0.796 to 0.902), recall+ (from 0.52 to 0.747), F-measure (from 0.833 to 0.901), and size (from 1381 to 1517). In terms of run time, 4 out of 11 systems computed an alignment in less than 100 seconds, a ratio which is similar to 2019 (5 out of 12). LogMapLite remains the system with the shortest runtime. Regarding quality, AML remains the system with the highest F-measure (0.941) and recall+ (0.81), but 3 other systems obtained an F-measure above 0.88 (Lily, LogMapBio, and LogMap) which is at least as good as the best systems in OAEI 2007-2010. Like in previous years, there is no significant correlation between the quality of the generated alignment and the run time. Four systems produced coherent alignments.

4.3 Biodiversity and Ecology

Four systems participating this year did participate to this track last year as well: AML and the LogMap family systems (LogMap, LogMapBio and LogMapLT). Three are new participants: ATBox, ALOD2Vec and Wiktionary. The newcomer ATBox did not register explicitly to the track but could cope with at least one task so we did include its results. As in the previous edition, we used precision, recall and F-measure to evaluate the performance of the participating systems. The results for the Biodiversity and Ecology track are shown in Table 9.

In comparison to previous years, we observed a decrease in the number of systems that succeeded to generate alignments for the ENVO-SWEET and FLOPO-PTO tasks. Basically, except of AML and the LogMap variants, only ATBox could cope with the

tasks with fair results. ALOD2Vec and Wiktionary generated a similar, huge set of non meaningful mappings with a very low F-measure as shown in Table 9.

Table 9. Results for the Biodiversity & Ecology track.

System	Time (s)	Number of mappings	Number of unique mappings	Precision	Recall	F-measure
FLOPO-PTO task						
LogMap	25.30	235	0	0.817	0.787	0.802
LogMapBio	450.71	236	1	0.814	0.787	0.800
AML	53.74	510	54	0.766	0.820	0.792
LogMapLt	17.02	151	0	0.987	0.611	0.755
ATBox	24.78	148	5	0.946	0.574	0.714
Wiktionary	1935	121.632	0	0.001	0.619	0.002
ALOD2Vec	246.37	121.633	1	0.001	0.619	0.002
ENVO-SWEET task						
AML	38.83	940	229	0.810	0.927	0.865
LogMapLt	32.70	617	41	0.904	0.680	0.776
ATBox	13.63	544	45	0.871	0.577	0.694
LogMap	35.15	440	0	0.964	0.516	0.672
LogMapBio	50.25	432	1	0.961	0.505	0.662
ANAEETHES-GEMET task						
LogMapBio	1243.15	397	0	0.924	0.876	0.899
LogMap	17.30	396	0	0.924	0.874	0.898
AML	4.17	328	24	0.976	0.764	0.857
LogMapLt	10.31	151	8	0.940	0.339	0.498
AGROVOC-NALT task						
AML	139.50	17.748	17.748	0.955	0.835	0.890

The results of the participating systems have slightly increased in terms of F-measure for both first two tasks compared to last year. In terms of run time, Wiktionary, ALOD2Vec and LogMapBio took the longer time, for the latter due to the loading of mediating ontologies from BioPortal.

For the FLOPO-PTO task, LogMap and LogMapBio achieved the highest F-measure. AML generated a large number of mappings (significantly bigger than the size of the reference alignment), those alignments were mostly subsumption ones. In order to evaluate the precision in a more significant manner, we had to calculate an approximation by manually assessing a subset of around 100 mappings, that were not present in the reference alignment. LogMapLt and ATBox achieved a high precision but the lowest recall.

Regarding the ENVO-SWEET task, AML ranked first in terms of F-measure, followed by LogMapLt and ATBox. The systems with the highest precision (LogMap and LogMapBio) achieve the lowest recall. Again here, AML generated a bigger set with a high number of subsumption mappings, it still achieved the best F-Measure for the

task. It is worth nothing that due the specific structure of the SWEET ontology, a lot of the false positives come from homonyms [45].

The ANAEETHES-GEMET and AGROVOC-NALT matching tasks have been introduced to the track this year, with the particularity of being resources developed in SKOS. Only AML could handle the files in their original format. LogMap and its variants could generate mappings for ANAEETHES-GEMET, based on ontology files after being transformed automatically into OWL. For the transformation, we made use of a source code[29] that was directly derived from AML ontology parsing module, kindly provided to us by its developers. LogMap and LogMapBio achieve the best results with LogMap processing the task in a shorter time. LogMapBio took a much longer time due to downloading 10 mediating ontologies from BioPortal, still the gain is not significant in terms of performance. The AGROVOC-NALT task has been managed only by AML. All other systems failed in generating mappings on both the SKOS and OWL versions of the thesauri. AML achieves good results and a very high precicion. It generated a higher number of mappings (around 1000 more) than the curated reference alignment. We performed a manual assessment of a subset of those mappings to reevaluate the precision and F-measure.

Overall, in this third evaluation, the results obtained from participating systems for the two tasks ENVO-SWEET and FLOPO-PTO remained similar with a slight increase in terms of F-measure compared to last year. The results of the two new tracks demonstrate systems (beside AML) are not ready to handle SKOS. Sometimes automatically transforming to OWL helps to avoid the issue, sometimes not. The number of mappings in the AGROVOC-NALT track is really a challenge and AML does not loose in performance which demonstrates that besides being the more tolerant tool in terms of format, it also scales up to large size thesauri.

4.4 Conference

The conference evaluation results using the sharp reference alignment *rar2* are shown in Table 10. For the sake of brevity, only results with this reference alignment and considering both classes and properties are shown. For more detailed evaluation results, please check conference track's web page.

With regard to two baselines we can group tools according to system's position: eight matching systems outperformed both baselines (ALIN, AML, ALOD2Vec, ATBox, LogMap, LogMapLt, VeeAlign and Wiktionary); two performed worse than both baselines (DESKMatcher and Lily). Three matchers (ALIN and Lily) do not match properties at all. Naturally, this has a negative effect on their overall performance.

The performance of all matching systems regarding their precision, recall and F_1-measure is plotted in Figure 1. Systems are represented as squares or triangles, whereas the baselines are represented as circles.

With respect to *logical coherence* [63, 64], as the last year, only three tools (ALIN, AML and LogMap) have no consistency principle violation.

As the last year we performed analysis of the *False Positives*, i.e. correspondences discovered by the tools which were evaluated as incorrect. The list of the False Positives

[29] `http://oaei.ontologymatching.org/2020/biodiv/code/SKOS2OWL.zip`

Table 10. The highest average $F_{[0.5|1|2]}$-measure and their corresponding precision and recall for each matcher with its F_1-optimal threshold (ordered by F_1-measure). Inc.Align. means number of incoherent alignments. Conser.V. means total number of all conservativity principle violations. Consist.V. means total number of all consistency principle violations.

System	Prec.	$F_{0.5}$-m.	F_1-m.	F_2-m.	Rec.	Inc.Align.	Conser.V.	Consist.V.
VeeAlign	0.74	0.72	0.7	0.67	0.66	9	76	83
AML	0.78	0.74	0.69	0.65	0.62	0	39	0
LogMap	0.77	0.72	0.66	0.6	0.57	0	25	0
Wiktionary	0.66	0.63	0.58	0.54	0.52	7	133	27
ATBox	0.58	0.58	0.57	0.56	0.56	10	192	52
LogMapLt	0.68	0.62	0.56	0.5	0.47	5	96	25
ALIN	0.82	0.69	0.56	0.48	0.43	0	2	0
ALOD2Vec	0.64	0.6	0.56	0.51	0.49	10	427	229
edna	0.74	0.66	0.56	0.49	0.45			
StringEquiv	0.76	0.65	0.53	0.45	0.41			
Lily	0.62	0.57	0.51	0.46	0.43	5	100	43
DESKMatcher	0.1	0.12	0.16	0.27	0.47	13	895	391

is available on the conference track's web page as well as further details about this evaluation. Comparing to the previous year we added the comparison of "why was an alignment discovered" assigned by us with the explanation for the alignment provided by the system itself. This year three systems generated explanations with the mappings ALOD2Vec, DESKMatcher and Wiktionary.

The Conference evaluation results using the *uncertain reference alignments* are presented in Table 11. Out of the 10 alignment systems, three (ALIN, DESKMatcher, LogMapLt) use 1.0 as the confidence value for all matches they identify. The remaining 7 systems (ALOD2Vec, AML, ATBOX, Lily, LogMap, VeeAlign, Wiktionary) have a wide variation of confidence values.

Table 11. F-measure, precision, and recall of the different matchers when evaluated using the sharp (*ra1*), discrete uncertain and continuous uncertain metrics.

	Sharp			Discrete			Continuous		
System	Prec	F-ms	Rec	Prec	F-ms	Rec	Prec	F-ms	Rec
ALIN	0.87	0.60	0.46	0.87	0.69	0.57	0.87	0.70	0.60
ALOD2Vec	0.69	0.59	0.52	0.81	0.67	0.58	0.70	0.65	0.60
AML	0.84	0.74	0.66	0.79	0.78	0.77	0.80	0.77	0.74
ATBOX	0.68	0.60	0.53	0.65	0.64	0.64	0.65	0.65	0.66
DESKMacther	0.11	0.18	0.50	0.11	0.18	0.63	0.11	0.18	0.63
Lily	0.67	0.56	0.47	1.00	0.01	0.01	0.64	0.31	0.20
LogMap	0.82	0.69	0.59	0.81	0.70	0.62	0.80	0.67	0.57
LogMapLt	0.73	0.59	0.50	0.73	0.67	0.62	0.72	0.67	0.63
VeeAlign	0.78	0.73	0.69	0.69	0.72	0.76	0.69	0.73	0.76
Wiktionary	0.70	0.61	0.54	0.79	0.55	0.42	0.74	0.60	0.51

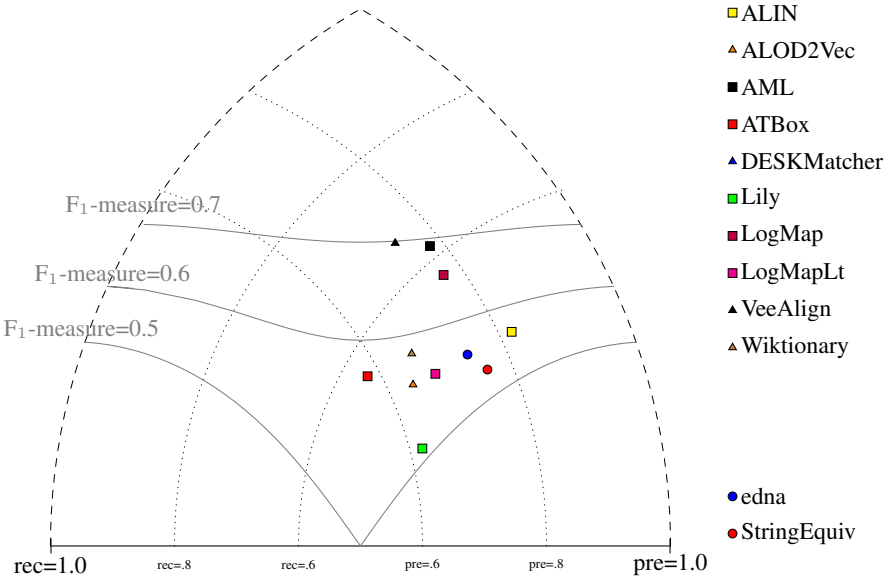


Fig. 1. Precision/recall triangular graph for the conference test case. Dotted lines depict level of precision/recall while values of F_1-measure are depicted by areas bordered by corresponding lines F_1-measure=0.[5|6|7].

When comparing the performance of the systems on the uncertain reference alignments versus that on the sharp version, we see that in the discrete case all systems except Lily performed the same or better in terms of F-measure (Lily's F-measure dropped almost to 0). Changes in F-measure of discrete cases ranged from -1 to 15 percent over the sharp reference alignment. This was predominantly driven by increased recall, which is a result of the presence of fewer 'controversial' matches in the uncertain version of the reference alignment.

The performance of the systems with confidence values always 1.0 is very similar regardless of whether a discrete or continuous evaluation methodology is used, because many of the matches they find are the ones that the experts had high agreement about, while the ones they missed were the more controversial matches. AML produces a fairly wide range of confidence values and has the highest F-measure under both the continuous and discrete evaluation methodologies, indicating that this system's confidence evaluation does a good job of reflecting cohesion among experts on this task. Of the remaining systems, three (ALOD2Vec, AML, LogMap) have relatively small drops in F-measure when moving from discrete to continuous evaluation. Lily's performance drops drastically under the discrete and continuous evaluation methodologies. This is because the system assigns low confidence values to some matches in which the labels are equivalent strings, which many crowdsourcers agreed with unless there was a compelling technical reason not to. This hurts recall significantly.

Overall, in comparison with last year, the F-measures of most returning matching systems essentially held constant when evaluated against the uncertain reference align-

ments. The exception was Lily, whose performance in discrete case decreased dramatically. ALOD2Vec, ATBOX, DESKMacther, VeeAlign are four new systems participating in this year. ALOD2Vec's performance increases 14 percent in discrete case and 11 percent in continuous case in terms of F-measure over the sharp reference alignment from 0.59 to 0.67 and 0.65 respectively, which it is mainly driven by increased recall. It is also interesting that the precision of ALOD2Vec increases 17 percent in discrete case over the sharp version. It is because ALOD2Vec assigns low confidence values to those pairs that don't have identical labels, which might help to remove some false positives in discrete case. ATBOX performs slightly better in both discrete and continuous cases compared to the sharp case in term of F-measure, which increases from 0.60 to 0.64 and 0.66 respectively. This is also mostly driven by increased recall. From the results, DESKMacther achieves low precision among three different versions of reference alignment in general because it assigns all matches with 1.0 confidence value even the labels of two entities have low string similarity. Reasonably, it achieves slightly better recall from sharp to discrete and continuous cases, while the precision and F-measure remain constant. VeeAlign's performance stays mostly constant from sharp to discrete and continuous in term of F-measure.

This year we conducted experiment of matching *cross-domain DBpedia ontology to OntoFarm ontologies*. In order to evaluate resulted alignments we prepared reference alignment of DBpedia to three OntoFarm ontologies (ekaw, sigkdd and confOf) as explained in [61]. This was not announced beforehand and systems did not specifically prepare for this. Out of 10 systems five managed to match DBpedia to OntoFarm ontologies (there were different problems dealing with parsing of the DBpedia ontology): AML, DESKMacther, LogMap, LogMapLt and Wiktionary.

We evaluated alignments from the systems and the results are in Table 12. Additionally, we added two baselines: StringEquiv as a string matcher based on string equality applied on local names of entities which were lowercased and edna as a string editing distance matcher.

Table 12. Threshold, F-measure, precision, and recall of systems when evaluated using reference alignment for DBpedia to OntoFarm ontologies

System	Thres.	Prec.	$F_0.5$-m.	F_1-m.	F_1-m.	Rec.
AML	0.81	0.48	0.51	0.56	0.62	0.67
edna	0.91	0.34	0.38	0.45	0.56	0.67
StringEquiv	0	0.32	0.35	0.42	0.51	0.6
Wiktionary	0.41	0.36	0.38	0.43	0.48	0.53
LogMap	0	0.37	0.39	0.41	0.45	0.47
LogMapLt	0	0.33	0.34	0.36	0.38	0.4
DESKMatcher	0	0	0	0	0	0

We can see the systems perform almost the same as two baselines except AML which dominates with 0.56 of F1-measure. Low scores of measures show that the corresponding test cases are difficult for traditional ontology matching systems since they

mainly focus on matching of domain ontologies. It is supposed to be announced as new test cases for the conference track within OAEI 2021.

4.5 Disease and Phenotype Track

In the OAEI 2020 phenotype track 7 systems were able to complete at least one of the tasks with a 6 hours timeout. Table 13 shows the evaluation results in the HP-MP and DOID-ORDO matching tasks, respectively.

Table 13. Results for the HP-MP and DOID-ORDO tasks based on the consensus reference alignment.

System	Time (s)	# Corresp.	# Unique	Scores			Incoherence	
				Prec.	F-m.	Rec.	Unsat.	Degree
HP-MP task								
LogMap	32	2,128	9	0.90	0.83	0.77	≥0	≥0.0%
LogMapBio	1,355	2,198	62	0.88	0.83	0.78	≥0	≥0.0%
AML	102	2,029	358	0.91	0.82	0.74	≥0	≥0.0%
LogMapLt	7	1,370	0	1.00	0.71	0.55	≥0	≥0.0%
ATBox	16	759	10	0.98	0.46	0.30	≥0	≥0.0%
ALOD2Vec	2,384	67,943	469	0.02	0.05	0.64	≥0	≥0.0%
Wiktionary	854	67,455	4	0.02	0.04	0.63	≥0	≥0.0%
DOID-ORDO task								
LogMapBio	2,034	2,584	147	0.95	0.75	0.63	≥0	≥0.0%
AML	200	4,781	195	0.68	0.75	0.83	≥0	≥0.0%
LogMap	25	2,330	0	0.99	0.74	0.59	≥0	≥0.0%
Wiktionary	858	7,336	5	0.48	0.63	0.90	≥3,288	≥24.1%
LogMapLt	8	1,747	10	0.99	0.61	0.44	≥0	≥0.0%
ALOD2Vec	2,809	7,805	457	0.45	0.61	0.91	≥12,787	≥93.6%
ATBox	21	1,318	17	0.99	0.50	0.33	≥0	≥0.0%

Since the consensus reference alignments only allow us to assess how systems perform in comparison with one another, the proposed ranking is only a reference. Note that some of the correspondences in the consensus alignment may be erroneous (false positives) because all systems that agreed on it could be wrong (e.g., in erroneous correspondences with equivalent labels, which are not that uncommon in biomedical tasks). In addition, the consensus alignments will not be complete, because there are likely to be correct correspondences that no system is able to find, and there are a number of correspondences found by only one system (and therefore not in the consensus alignments) which may be correct. Nevertheless, the results with respect to the consensus alignments do provide some insights into the performance of the systems.

Overall, LogMap, LogMapBio and AML are the systems that provide the closest set of correspondences to the consensus (not necessarily the best system) in both tasks. LogMap has a small set of unique correspondences as most of its correspondences are also suggested by its variant LogMapBio and vice versa. Wiktionary and ALOD2Vec suggest a very large number of correspondences in the HP-MP task with respect to the

Table 14. Results for the whole ontologies matching tasks in the OAEI largebio track.

System	Time (s)	# Corresp.	# Unique	Scores			Incoherence	
				Prec.	F-m.	Rec.	Unsat.	Degree
Whole FMA and NCI ontologies (Task 2)								
AML	82	3,109	442	0.81	0.84	0.88	2	0.013%
LogMap	9	2,668	33	0.87	0.84	0.81	3	0.019%
LogMapBio	1,447	2,855	88	0.83	0.83	0.83	2	0.013%
LogMapLt	9	3,458	70	0.68	0.74	0.82	5,554	36.1%
Wiktionary	14,136	4,067	507	0.60	0.71	0.86	8,128	52.8%
ATBox	41	2,807	265	0.70	0.69	0.69	9,313	60.5%
Whole FMA ontology with SNOMED large fragment (Task 4)								
LogMapBio	7,046	6,470	162	0.83	0.73	0.65	0	0.0%
LogMap	624	6,540	271	0.81	0.72	0.64	0	0.0%
AML	181	8,163	2,818	0.69	0.70	0.71	0	0.0%
Wiktionary	24,379	2,034	227	0.78	0.34	0.22	989	3.0%
LogMapLt	15	1,820	26	0.85	0.33	0.21	974	2.9%
ATBox	54	1,880	124	0.80	0.33	0.21	958	2.9%
Whole NCI ontology with SNOMED large fragment (Task 6)								
AML	381	14,196	2,209	0.86	0.77	0.69	$\geq$535	$\geq$0.6%
LogMap	719	13,230	105	0.87	0.75	0.65	$\geq$1	$\geq$0.001%
LogMapBio	4,069	13,495	929	0.83	0.71	0.63	$\geq$0	$\geq$0.0%
LogMapLt	18	12,864	525	0.80	0.66	0.57	$\geq$72,865	$\geq$87.1%
Wiktionary	18,361	13,668	1,188	0.77	0.66	0.58	$\geq$68,466	$\geq$81.8%
ATBox	75	10,621	245	0.87	0.64	0.51	$\geq$65,543	$\geq$78.3%

other systems which suggest that it may also include many subsumption and related correspondences and not only equivalence. All systems produce coherent alignments except for Wiktionary and ALOD2Vec in the DOID-ORDO task.

4.6 Large Biomedical Ontologies

In the OAEI 2020 Large Biomedical Ontologies track, 8 systems were able to complete at least one of the tasks within a 6 hours timeout. Six systems were able to complete all six tasks.[30] The evaluation results for the largest matching tasks are shown in Table 14.

The top-ranked systems by F-measure were respectively: AML and LogMap in Task 2; LogMapBio and LogMap in Task 4; and AML and LogMap in Task 6. Interestingly, the use of background knowledge led to an improvement in recall from LogMapBio over LogMap in Tasks 2 and 4, but this came at the cost of precision, resulting in the two variants of the system having very similar F-measures.

The effectiveness of all systems decreased from small fragments to whole ontologies tasks.[31] One reason for this is that with larger ontologies there are more plausible

[30] Check out the supporting scripts to reproduce the evaluation: `https://github.com/ernestojimenezruiz/oaei-evaluation`

[31] `http://www.cs.ox.ac.uk/isg/projects/SEALS/oaei/2020/results/`

correspondence candidates, and thus it is harder to attain both a high precision and a high recall. In fact, this same pattern is observed moving from the FMA-NCI to the FMA-SNOMED to the SNOMED-NCI problem, as the size of the task also increases. Another reason is that the very scale of the problem constrains the matching strategies that systems can employ: AML for example, forgoes its matching algorithms that are computationally more complex when handling very large ontologies, due to efficiency concerns. The size of the whole ontologies tasks proved a problem for a some of the systems, which were unable to complete them within the allotted time: ALOD2Vec and DESKMatcher.

With respect to alignment coherence, as in previous OAEI editions, only two distinct systems have shown alignment repair facilities: AML, LogMap and its LogMapBio variant. Note that only LogMap and LogMapBio are able to reduce to a minimum the number of unsatisfiable classes across all tasks, missing 3 unsatisfiable classes in the worst case (whole FMA-NCI task). As the results tables show, even the most precise alignment sets may lead to a huge number of unsatisfiable classes. This proves the importance of using techniques to assess the coherence of the generated alignments if they are to be used in tasks involving reasoning. We encourage ontology matching system developers to develop their own repair techniques or to use state-of-the-art techniques such as Alcomo [52], the repair module of LogMap (LogMap-Repair) [41] or the repair module of AML [60], which have worked well in practice [43, 29].

4.7 Multifarm

This year, 6 systems registered to participate in the MultiFarm track: AML, Lily, LogMap, LogMapLT, Wiktionary and VeeAlign. This number slightly increases with respect to the last campaign (5 in 2019, 6 in 2018, 8 in 2017, 7 in 2016, 5 in 2015, 3 in 2014, 7 in 2013, and 7 in 2012). Lily has generated empty alignments so there are no results to be reported.

The tools heavily rely on the lexical matching approach with the exception of VeeAlign system which adopts a deep learning approach. *VeeAlign* uses a supervised deep learning approach to discover alignments proposing a two-step model with multi-facted context representation to produce contextualised representations of concepts, which aids alignment based on semantic and structural properties of an ontology. *AML* employs lexical matching techniques using a translation module, with an emphasis on the use of background knowledge.The tool also includes structural components for both matching and filtering steps and features a logical repair algorithm. *Lily* matcher measures the literal similarity between ontologies on the extracted semantic subgraph and follows structure-based methods, background knowledge and document matching technologies. *Logmap* uses a lexical inverted index to compute the initial set of mappings which are then supported by logic based extractions with built-in reasoning and repair diagnosis capabilities. On the other hand *LogMapLt* (Logmap "lightweight") essentially only applies (efficient) string matching techniques for a lightweigh and fast computation. *Wiktionary* matcher is based on an online lexical resource, namely Wiktionary but also utilizes the schema matching and produces an explanation for the discovered correspondence. The reader can refer to the OAEI papers for a detailed description of the strategies adopted by each system.

The Multifarm evaluation results based on the blind dataset are presented in Table 15. They have been computed using the Alignment API 4.9 and can slightly differ from those computed with the SEALS client. We haven't applied any threshold on the results. We do not report the results of non-specific systems here, as we could observe in the last campaigns that they can have intermediate results in the "same ontologies" task (ii) and poor performance in the "different ontologies" task (i). The detailed results can be investigated on the page of multifarm track results[32].

Table 15. MultiFarm aggregated results per matcher, for each type of matching task – different ontologies (i) and same ontologies (ii). Time is measured in minutes (for completing the 55×24 matching tasks) – ** tool run in a different environment so runtime is not reported; #pairs indicates the number of pairs of languages for which the tool is able to generate (non-empty) alignments; size indicates the average of the number of generated correspondences for the tests where an (non-empty) alignment has been generated. Two kinds of results are reported: those not distinguishing empty and erroneous (or not generated) alignments and those—indicated between parenthesis—considering only non-empty generated alignments for a pair of languages.

System	Time	#pairs	Type (i) – 22 tests per pair				Type (ii) – 2 tests per pair			
			Size	Prec.	F-m.	Rec.	Size	Prec.	F-m.	Rec.
AML	170	55	8.25	.72 (.72)	.47 (.47)	.35 (.35)	33.65	.94 (.96)	.28 (.28)	.17 (.17)
LogMap	43	55	6.64	.73 (.72)	.37 (.37)	.25 (.25)	46.62	.95 (.97)	.42 (.43)	.28 (.28)
LogMapLt	17	23	1.15	.34 (.35)	.04 (.09)	.02 (.02)	95.17	.02 (.02)	.01 (.03)	.01 (.01)
VeeAlign	**	54	2.53	.73 (.77)	.15 (.15)	.09 (.09)	11.98	.91 (.93)	.14 (.14)	.08 (.08)
Wiktionary	1290	53	4.92	.77 (.80)	.32 (.33)	.21 (.21)	9.38	.94 (.96)	.12 (.13)	.07 (.07)

AML outperforms all other systems in terms of F-measure for task i) (same behaviour in the last campaigns). In terms of precision, Wiktionary is the system that generates the most precise alignments, followed by LogMap, VeeAlign and AML. With respect to the task ii) LogMap has the overall best performance. Comparing the results from last year, in terms F-measure (cases of type i), AML maintains its overall performance (.45 in 2019, .46 in 2018, .46 in 2017, .45 in 2016 and .47 in 2015). The same could be observed for LogMap (.37 in 2019, .37 in 2018, .36 in 2017, and .37 in 2016). The performance in terms of F-measure of Wiktionary also remains stable. In terms of runtime, the results are not really comparable with the ones in the last campaign considering the fact the SEALS repositories have been moved to another server with a different configuration.

Overall, the F-measure for blind tests remains relatively stable across campaigns. As observed in previous campaigns, systems still privilege precision over recall. Furthermore, the overall results in MultiFarm are lower than the ones obtained for the original English version of the Conference dataset.

[32] `http://oaei.ontologymatching.org/2020/results/multifarm/index.html`

4.8 Link Discovery

This year the Link Discovery track counted three participants in the Spatial test case: AML, Silk and RADON. Those were the exact same systems (and versions) that participated on OAEI 2019.

We divided the Spatial test cases into four suites. In the first two suites (SLL and LLL), the systems were asked to match LineStrings to LineStrings considering a given relation for 200 and 2K instances for the TomTom and Spaten datasets. In the last two tasks (SLP, LLP), the systems were asked to match LineStrings to Polygons (or Polygons to LineStrings depending on the relation) again for both datasets. Since the precision, recall and F-measure results from all systems were equal to 1.0, we are only presenting results regarding the time performance. The time performance of the matching systems in the SLL, LLL, SLP and LLP suites are shown in Figures 2-3. The results can also be found in HOBBIT git (`https://hobbit-project.github.io/OAEI_2020.html`).

In the SLL suite, RADON has the best performance in most cases except for the *Touches* and *Intersects* relations, followed by AML. Silk seems to need the most time, particularly for *Touches* and *Intersects* relations in the TomTom dataset and *Overlaps* in both datasets.

In the LLL suite we have a more clear view of the capabilities of the systems with the increase in the number of instances. In this case, RADON and Silk have similar behavior as in the small dataset, but it is more clear that the systems need much more time to match instances from the TomTom dataset. RADON has still the best performance in most cases. AML has the next best performance and is able to handle some cases better than other systems (e.g. *Touches* and *Intersects*), however, it also hits the platform time limit in the case of *Disjoint*.

In the SLP suite, in contrast to the first two suites, RADON has the best performance for all relations. AML and Silk have minor time differences and, depending on the case, one is slightly better than the other. All the systems need more time for the TomTom dataset but due to the small size of the instances the time difference is minor.

In the LLP suite, RADON again has the best performance in all cases. AML hits the platform time limit in *Disjoint* relations on both datasets and is better than Silk in most cases except *Contains* and *Within* on the TomTom dataset where it needs an excessive amount of time.

Taking into account the executed test cases we can identify the capabilities of the tested systems as well as suggest some improvements. All the systems participated in most of the test cases, with the exception of Silk which did not participate in the *Covers* and *Covered By* test cases.

RADON was the only system that successfully addressed all the tasks, and had the best performance for the SLP and LLP suites, but it can be improved for the *Touches* and *Intersects* relations for the SLL and LLL suites. AML performs extremely well in most cases, but can be improved in the cases of *Covers/Covered By* and *Contains/Within* when it comes to LineStrings/Polygons Tasks and especially in *Disjoint* relations where it hits the platform time limit. Silk can be improved for the *Touches*, *Intersects* and *Overlaps* relations and for the SLL and LLL tasks and for the *Disjoint* relation in SLP and LLP Tasks.

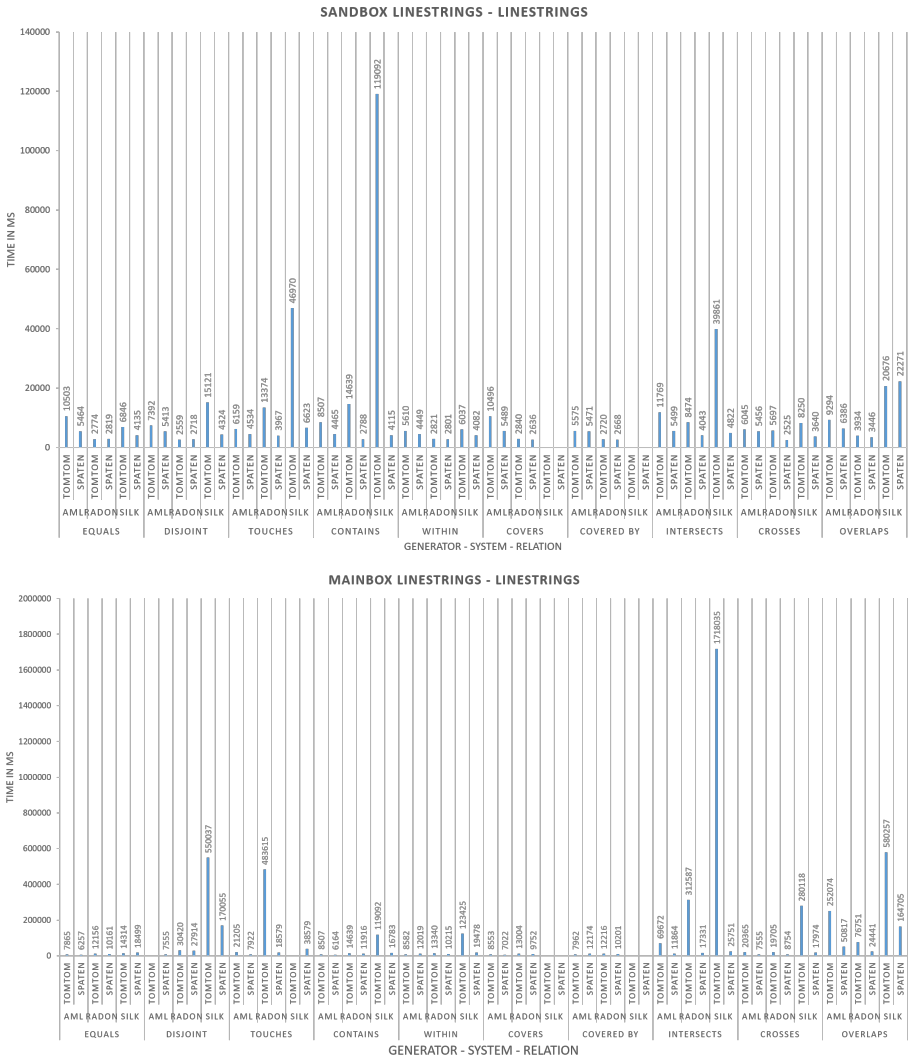


Fig. 2. Time performance for TomTom & Spaten SLL (top) and LLL (bottom) suites for AML (A), Silk (S) and RADON (R).

In general, all systems needed more time to match the TomTom dataset than the Spaten one, due to the smaller number of points per instance in the latter. Comparing the LineString/LineString to the LineString/Polygon Tasks we can say that all the systems needed less time for the first for the *Contains*, *Within*, *Covers* and *Covered by* relations, more time for the *Touches*, *Instersects* and *Crosses* relations, and approximately the same time for the *Disjoint* relation.

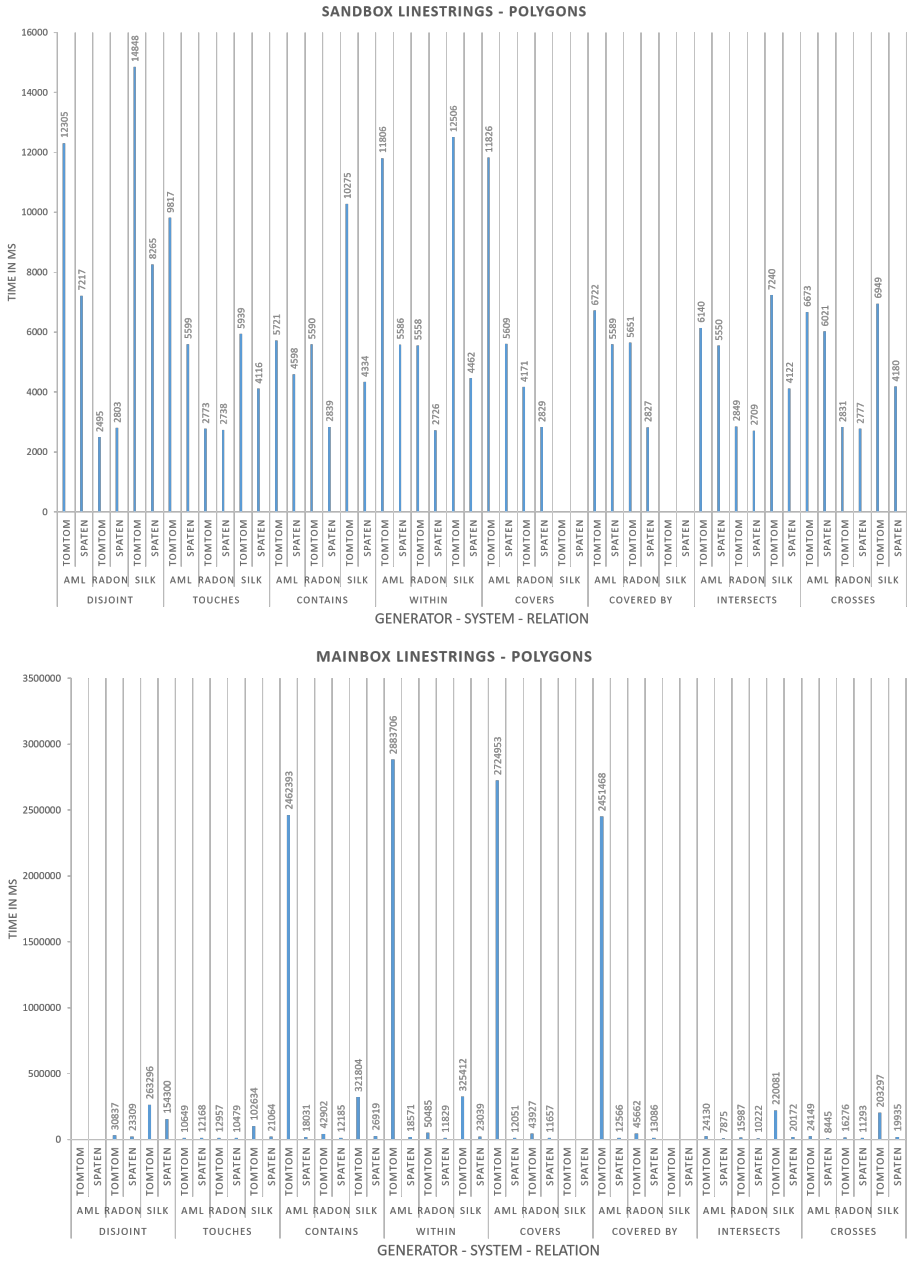


Fig. 3. Time performance for TomTom & Spaten SLP (top) and LLP (bottom) suites for AML (A), Silk (S) and RADON (R).

4.9 SPIMBENCH

This year, the SPIMBENCH track counted five participants: AML, Lily, LogMap, FTRLIM and REMiner. REMiner participated for the first time this year while AML, Lily, LogMap and FTRLIM also participated last year. The evaluation results of the track are shown in Table 16. The results can also be found in HOBBIT git (`https://hobbit-project.github.io/OAEI_2020.html`).

Table 16. Results for SPIMBENCH task.

Sandbox Dataset (380 instances, 10000 triples)				
System	**Fmeasure**	**Precision**	**Recall**	**Time (in ms)**
LogMap	0.8413	0.9382	0.7625	7483
AML	0.8645	0.8348	0.8963	6446
Lily	0.9917	0.9835	1	2050
FTRLIM	0.9214	0.8542	1	1525
REMiner	0.9983	1	0.9966	7284
Mainbox Dataset (1800 instances, 50000 triples)				
System	**Fmeasure**	**Precision**	**Recall**	**Time (in ms)**
LogMap	0.7856	0.8801	0.7094	26782
AML	0.8604	0.8385	0.8835	38772
Lily	0.9953	0.9908	1	3899
FTRLIM	0.9214	0.8558	0.9980	2247
REMiner	0.9976	0.9986	0.9966	33966

Lily and FTRLIM had the best performance overall both in terms of F-measure and run time. Notably, their run time scaled very well with the increase in the number of instances. REMiner produces the best results (almost full) for all metrics. Lily, FTRLIM and AML had a higher recall than precision, while Lily and FTRLIM had a full recall. By contrast, REMiner and LogMap had a higher precision and lower recall, while REMiner had a full precision. AML, LogMap and REMiner had a similar run time performance.

4.10 Geolink Cruise

We evaluated all participants in the OAEI 2020. Unfortunately, none of the current alignment systems can generate the coreferences between the cruise instances in the Geolink Cruise benchmark. The state of the art alignment systems work well on finding the links with a higher string similarity or string synonyms between two objects. However, in terms of the instances with lower string similarities, or the external information is not available or very limited to help the aligning task. Another kind of algorithm is needed, like finding the relation of the instances based on the underlying structure of the graphs. We hope that system will manage this track in future years.

4.11 Knowledge Graph

We evaluated all SEALS participants in the OAEI (even those not registered for the track) on a very small matching task[33]. This revealed that not all systems were able to handle the task, and in the end, only the following systems were evaluated: ALOD2Vec, AML, ATBox, DESKMatcher, LogMapKG, LogMapLt, Wiktionary. We also evaluated LogMapBio but compared to LogMapKG it does not change the results (meaning that the external knowledge does not help in these cases which is reasonable). LogMapKG is the LogMap systems which returns TBox as well as ABox correspondences. In this year, two systems registered especially for this track but were unable to finally submit their system in time. This shows that there is a demand for this track and we plan to provide this track also next year. We hope that the system developers are able to submit the system next year. In comparison to the previous years, we have new matchers like ALOD2Vec (which produced an error in 2018), ATBox (new), and DESKMatcher (new).

What did not change over the years is that some matchers do not return a valid alignment file. The reason is the xml format of this file together with URIs in the knowledge graph containing special characters e.g. ampersand. These characters should be encoded, in order that xml parsers can process this file. Thus a post processing step is executed which tries to create a valid xml file. The resulting alignments are available for download. [34]

Table 17 shows the aggregated results for all systems, including the number of tasks in which they were able to generate a non-empty alignment (#tasks) and the average number of generated correspondences in those tasks (size). We report the macro averaged precision, F-measure, and recall results where we do not distinguishing empty and erroneous (or not generated) alignments. The values between parentheses show the results when considering only non empty alignments.

All systems were able to generate class correspondences. In terms of F-measure, AML is still the best one and only DESKMatcher could not beat the baselines. The recall values are higher than last year (maximum of 0.77) which shows that some matchers improved and can find more class correspondences. Nevertheless there is still room for improvement and some of these class matches looks like they are not easy to find.

In the third year of this track all systems except the LogMap family are able to return property correspondences. This is a huge improvement (which happens over the years) because it makes the systems more usable in real case scenarios where a property might not be classified as owl:ObjectProperty or owl:DatatypeProperty. The systems ALOD2Vec, ATBox, and Wiktionary could achieve a F-measure of 0.95 or more which shows that property matching is easier in this track than class or instance matching.

With respect to instance correspondences, two systems (ALOD2Vec and Wiktionary) exceed the best performance of last year with an F-measure of 0.87. The margin between the baseline and the best systems is now a bit greater but still only 0.03 away. Again LogMapKG returns a much higher number of instance correspondences (29,190

[33] http://oaei.ontologymatching.org/2019/results/knowledgegraph/small_test.zip

[34] http://oaei.ontologymatching.org/2020/results/knowledgegraph/oaei2020-knowledgegraph-alignments.zip

Table 17. Knowledge Graph track results, divided into class, property, instance, and overall performance. For matchers that were not capable to complete all tasks, the numbers in parantheses denote the performance when only averaging across tasks that were completed.

System	Time (s)	# tasks	Size	Prec.	F-m.	Rec.
Class performance						
ALOD2Vec	0:13:24	5	20.0	1.00	0.80	0.67
AML	0:50:55	5	23.6	0.98	0.89	0.81
ATBox	0:16:22	5	25.6	0.97	0.87	0.79
baselineAltLabel	0:10:57	5	16.4	1.00	0.74	0.59
baselineLabel	0:10:44	5	16.4	1.00	0.74	0.59
DESKMatcher	0:13:54	5	91.4	0.76	0.71	0.66
LogMapKG	2:47:51	5	24.0	0.95	0.84	0.76
LogMapLt	0:07:19	4	23.0	0.80 (1.00)	0.56 (0.70)	0.43 (0.54)
Wiktionary	0:30:12	5	22.4	1.00	0.80	0.67
Property performance						
ALOD2Vec	0:13:24	5	76.8	0.94	0.95	0.97
AML	0:50:55	5	48.4	0.92	0.70	0.57
ATBox	0:16:22	5	78.8	0.97	0.96	0.95
baselineAltLabel	0:10:57	5	47.8	0.99	0.79	0.66
baselineLabel	0:10:44	5	47.8	0.99	0.79	0.66
DESKMatcher	0:13:54	5	0.0	0.00	0.00	0.00
LogMapKG	2:47:51	5	0.0	0.00	0.00	0.00
LogMapLt	0:07:19	4	0.0	0.00	0.00	0.00
Wiktionary	0:30:12	5	80.0	0.94	0.95	0.97
Instance performance						
ALOD2Vec	0:13:24	5	4893.8	0.91	0.87	0.83
AML	0:50:55	5	6802.8	0.90	0.85	0.80
ATBox	0:16:22	5	4858.8	0.89	0.84	0.80
baselineAltLabel	0:10:57	5	4674.8	0.89	0.84	0.80
baselineLabel	0:10:44	5	3641.8	0.95	0.81	0.71
DESKMatcher	0:13:54	5	3820.6	0.94	0.82	0.74
LogMapKG	2:47:51	5	29190.4	0.40	0.54	0.86
LogMapLt	0:07:19	4	6653.8	0.73 (0.91)	0.67 (0.84)	0.62 (0.78)
Wiktionary	0:30:12	5	4893.8	0.91	0.87	0.83
Overall performance						
ALOD2Vec	0:13:24	5	4990.6	0.91	0.87	0.83
AML	0:50:55	5	6874.8	0.90	0.85	0.80
ATBox	0:16:22	5	4963.2	0.89	0.85	0.81
baselineAltLabel	0:10:57	5	4739.0	0.89	0.84	0.80
baselineLabel	0:10:44	5	3706.0	0.95	0.81	0.71
DESKMatcher	0:13:54	5	3912.0	0.93	0.81	0.72
LogMapKG	2:47:51	5	29214.4	0.40	0.54	0.84
LogMapLt	0:07:19	4	6676.8	0.73 (0.92)	0.66 (0.83)	0.61 (0.76)
Wiktionary	0:30:12	5	4996.2	0.91	0.87	0.83

in average) than all other participants but the recall is only slighly higher (0.03 to the next best recall of 0.83).

When analyzing the confidence values of the alignments, it turns out that most matchers makes use of the range between zero and one. Only DESKMatcher, LogMapLt, and the baselines return only 1.0. Further analysis can be made by browsing to the dashboard [35] which is generated with the MELT framework [37].

Regarding runtime, LogMapKG was was the slowest system (2:47:51 for all test cases), followed by AML (0:50:55). Besides the baseline, four matchers were able to compute the alignment in under 20 minutes which is a reasonable time for this track.

In this year we also run the matchers in the hidden test cases to see how many instance correspondences they return. The systems DESKMatcher, LogMapKG, and AML (in test case starwars-lyrics) run into memory issues. Due to the fact that there is no partial nor full gold standard available for these test cases, only the number of returned instances correspondences is analyzed. In [35] we run the matchers from OAEI 2019 on these hidden test cases and manually evaluated 1,050 returned correspondences. This results in the number of matches and a approximation of the precision for each matcher and test case. Based on these values, the estimated number of true positives for each test case can be calculated. The average and maximum number of expected instance correspondences is shown in table 18 together with the number of instance correspondences returned from OAEI 2020 matchers One can see that they return 1-2 orders of magnitude more correspondences than the number of expected true positives. Especially LogMapLt returns the highest number of correspondences in the first two test cases and Wiktionary in the last test case. ATBox and AML return less correspondences and a higher precision is expected in these test cases.

Table 18. Number of instance correspondences when matching the source wiki to the lyrics wiki.

source wiki	average	max	ALOD2Vec	AML	ATBox	LogMapLt	Wiktionary
marvelcinematicuniverse	292.7	584.8	1,175	1,052	987	2,403	1,175
memoryalpha	73.6	285.5	4,546	2,106	2,817	7,195	4,547
starwars	48.5	109.1	5,697	-	3,550	2,725	5,697

4.12 Interactive matching

This year, three systems participated in the Interactive matching track. They are ALIN, AML, and LogMap. Their results are shown in Table 19 and Figure 4 for both Anatomy and Conference datasets.

The table includes the following information (column names within parentheses):

– The performance of the system: Precision (Prec.), Recall (Rec.) and F-measure (F-m.) with respect to the fixed reference alignment, as well as Recall+ (Rec.+) for the

[35] `http://oaei.ontologymatching.org/2020/results/knowledgegraph/knowledge_graph_dashboard.html`

Table 19. Interactive matching results for the Anatomy and Conference datasets.

Tool	Error	Prec.	Rec.	F-m.	Rec.+	Prec. oracle	Rec. oracle	F-m. oracle	Tot. Reqs.	Dist. Mapps	Pos. Prec.	Neg. Prec.
Anatomy Dataset												
ALIN	NI	0.986	0.72	0.832	0.382	–	–	–	–	–	–	–
	0.0	0.988	0.856	0.917	0.623	0.988	0.856	0.917	360	953	1.0	1.0
	0.1	0.937	0.841	0.887	0.596	0.988	0.86	0.919	342	885	0.727	0.966
	0.2	0.895	0.827	0.86	0.57	0.989	0.862	0.921	337	872	0.553	0.929
	0.3	0.854	0.812	0.832	0.546	0.989	0.864	0.922	333	854	0.419	0.883
AML	NI	0.956	0.927	0.941	0.81	–	–	–	–	–	–	–
	0.0	0.972	0.933	0.952	0.822	0.972	0.933	0.952	189	189	1.0	1.0
	0.1	0.962	0.929	0.945	0.813	0.972	0.932	0.952	192	190	0.72	0.967
	0.2	0.951	0.928	0.939	0.809	0.972	0.935	0.954	212	210	0.529	0.933
	0.3	0.942	0.924	0.933	0.805	0.973	0.935	0.954	218	212	0.473	0.878
LogMap	NI	0.916	0.846	0.88	0.593	–	–	–	–	–	–	–
	0.0	0.988	0.846	0.912	0.595	0.988	0.846	0.912	388	1164	1.0	1.0
	0.1	0.967	0.831	0.894	0.567	0.971	0.803	0.879	388	1164	0.748	0.966
	0.2	0.95	0.82	0.881	0.549	0.952	0.765	0.848	388	1164	0.574	0.925
	0.3	0.938	0.818	0.874	0.543	0.927	0.723	0.812	388	1164	0.429	0.876
Conference Dataset												
ALIN	NI	0.874	0.456	0.599	–	–	–	–	–	–	–	–
	0.0	0.915	0.705	0.796	–	0.915	0.705	0.796	233	608	1.0	1.0
	0.1	0.75	0.679	0.713	–	0.928	0.736	0.821	232	597	0.581	0.988
	0.2	0.612	0.648	0.629	–	0.938	0.763	0.842	230	590	0.356	0.969
	0.3	0.516	0.617	0.562	–	0.945	0.783	0.856	227	579	0.239	0.946
AML	NI	0.841	0.659	0.739	–	–	–	–	–	–	–	–
	0.0	0.91	0.698	0.79	–	0.91	0.698	0.79	221	220	1.0	1.0
	0.1	0.843	0.682	0.754	–	0.916	0.714	0.803	242	237	0.714	0.965
	0.2	0.777	0.677	0.723	–	0.925	0.735	0.819	267	255	0.567	0.945
	0.3	0.721	0.65	0.684	–	0.929	0.742	0.825	270	253	0.452	0.879
LogMap	NI	0.818	0.59	0.686	–	–	–	–	–	–	–	–
	0.0	0.886	0.61	0.723	–	0.886	0.61	0.723	82	246	1.0	1.0
	0.1	0.851	0.6	0.703	–	0.858	0.574	0.688	82	246	0.703	0.983
	0.2	0.821	0.59	0.686	–	0.832	0.547	0.66	82	246	0.506	0.946
	0.3	0.804	0.585	0.677	–	0.817	0.522	0.637	82	246	0.385	0.909

NI stands for non-interactive, and refers to the results obtained by the matching system in the original track.

Anatomy task. To facilitate the assessment of the impact of user interactions, we also provide the performance results from the original tracks, without interaction (line with Error NI).

– To ascertain the impact of the oracle errors, we provide the performance of the system with respect to the oracle (i.e., the reference alignment as modified by the errors introduced by the oracle: Precision oracle (Prec. oracle), Recall oracle (Rec. oracle) and F-measure oracle (F-m. oracle). For a perfect oracle these values match the actual performance of the system.
– Total requests (Tot Reqs.) represents the number of distinct user interactions with the tool, where each interaction can contain one to three conflicting correspondences, that could be analysed simultaneously by a user.
– Distinct correspondences (Dist. Mapps) counts the total number of correspondences for which the oracle gave feedback to the user (regardless of whether they were submitted simultaneously, or separately).
– Finally, the performance of the oracle itself with respect to the errors it introduced can be gauged through the positive precision (Pos. Prec.) and negative precision (Neg. Prec.), which measure respectively the fraction of positive and negative answers given by the oracle that are correct. For a perfect oracle these values are equal to 1 (or 0, if no questions were asked).

The figure shows the time intervals between the questions to the user/oracle for the different systems and error rates. Different runs are depicted with different colors.

The matching systems that participated in this track employ different user-interaction strategies. While LogMap, and AML make use of user interactions exclusively in the post-matching steps to filter their candidate correspondences, ALIN can also add new candidate correspondences to its initial set. LogMap and AML both request feedback on only selected correspondences candidates (based on their similarity patterns or their involvement in unsatisfiabilities) and AML presents one correspondence at a time to the user. ALIN and LogMap can both ask the oracle to analyze several conflicting correspondences simultaneously.

The performance of the systems usually improves when interacting with a perfect oracle in comparison with no interaction. ALIN is the system that improves the most, because its high number of oracle requests and its non-interactive performance was the lowest of the interactive systems, and thus the easiest to improve.

Although system performance deteriorates when the error rate increases, there are still benefits from the user interaction—some of the systems' measures stay above their non-interactive values even for the larger error rates. Naturally, the more a system relies on the oracle, the more its performance tends to be affected by the oracle's errors.

The impact of the oracle's errors is linear for ALIN, and AML in most tasks, as the F-measure according to the oracle remains approximately constant across all error rates. It is supra-linear for LogMap in all datasets.

Another aspect that was assessed, was the response time of systems, i.e., the time between requests. Two models for system *response times* are frequently used in the literature [16]: Shneiderman and Seow take different approaches to categorize the response times taking a task-centered view and a user-centered view respectively. According to task complexity, Shneiderman defines response time in four categories: typing, mouse

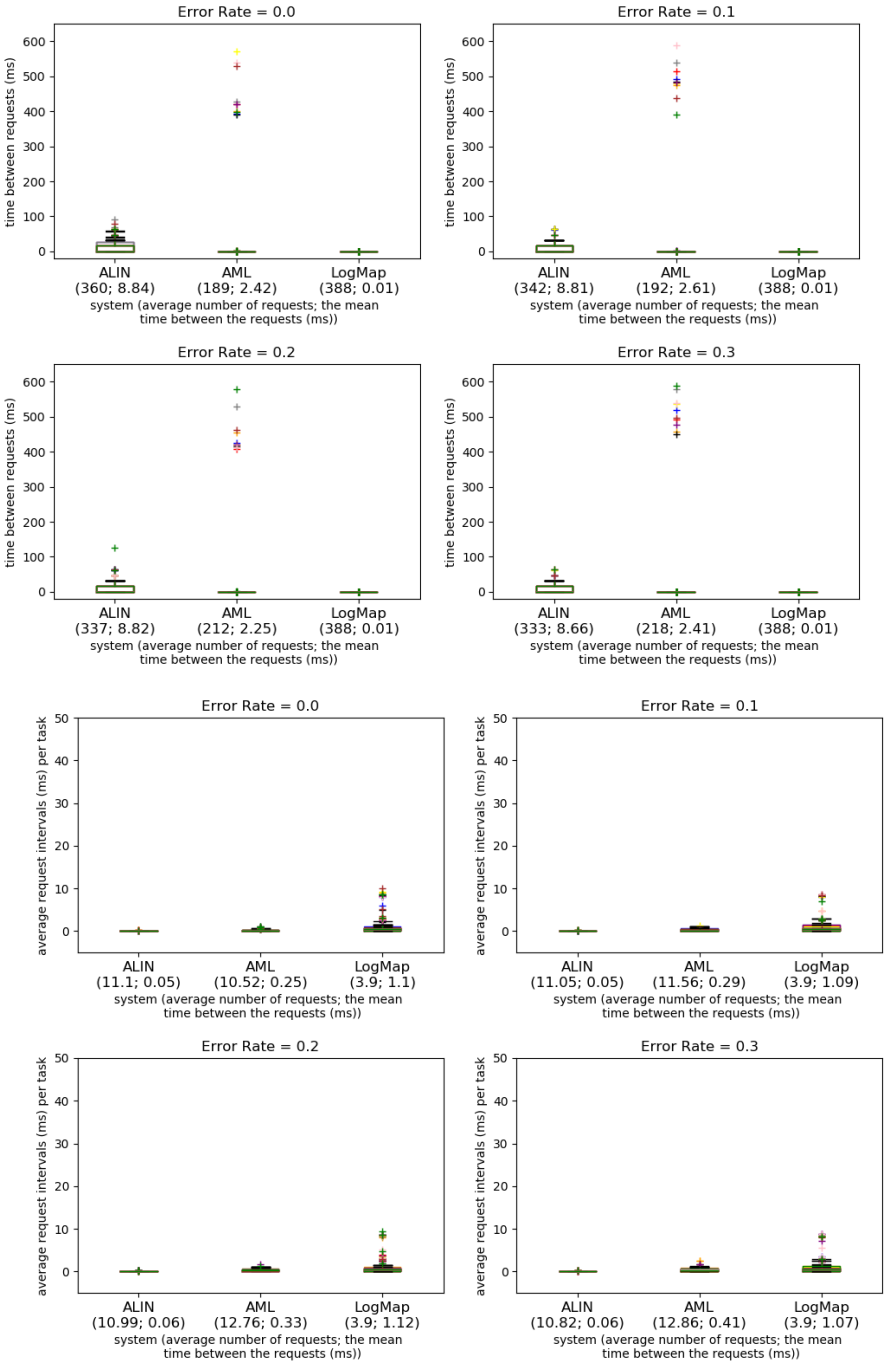


Fig. 4. Time intervals between requests to the user/oracle for the Anatomy (top 4 plots) and Conference (bottom 4 plots) datasets. Whiskers: Q1-1,5IQR, Q3+1,5IQR, IQR=Q3-Q1. The labels under the system names show the average number of requests and the mean time between the requests for the ten runs.

movement (50-150 ms), simple frequent tasks (1 s), common tasks (2-4 s) and complex tasks (8-12 s). While Seow's definition of response time is based on the user expectations towards the execution of a task: instantaneous (100-200 ms), immediate (0.5-1 s), continuous (2-5 s), captive (7-10 s). Ontology alignment is a cognitively demanding task and can fall into the third or fourth categories in both models. In this regard the response times (request intervals as we call them above) observed in all datasets fall into the tolerable and acceptable response times, and even into the first categories, in both models. The request intervals for AML, LogMap and ALIN stay at a few milliseconds for most datasets. It could be the case, however, that a user would not be able to take advantage of these low response times because the task complexity may result in higher user response time (i.e., the time the user needs to respond to the system after the system is ready).

4.13 Complex Matching

Table 20. Results of the Complex Track in OAEI 2020. Populated datasets (*Pop.*) using the metrics: precision (*Prec.*), coverage (*Cov.*), relaxed precision (*R_P*), relaxed recall (*R_R*) and relaxed f-measure (*R_F*).

Matcher	Pop. Conference		Hydrography			GeoLink			Pop. GeoLink			Pop. Enslaved			Taxon	
	Prec.	Cov.	R_P	R_F	R_R	R_P	R_F	R_R	R_P	R_F	R_R	R_P	R_F	R_R	Prec.	Cov.
ALIN	**.68-.98**	.20-.28	-	-	-	-	-	-	-	-	-	-	-	-	-	-
ALOD2Vec	.39-.78	.24-.33	-	-	-	-	-	-	-	-	-	-	-	-	.79-.96	.08-.14
AML	.59-.93	.31-.37	-	-	-	-	-	-	-	-	-	-	-	-	-	-
AMLC	.23-.51	.26-.31	.45	.10	.05	.50	.23	.23	.50	.32	.23	.73	.40	.28	.19-.40	0
AROA	-	-	-	-	-	-	-	-	.87	**.60**	**.46**	**.80**	**.51**	**.38**	-	-
ATBox	.39-.81	.27-.36	-	-	-	-	-	-	-	-	-	-	-	-	.56-.71	.06-.11
CANARD	.25-.88	**.40-.50**	-	-	-	-	-	-	**.89**	.54	.39	.42	.19	.13	.16-.57	**.17-.36**
LogMap	.56-.96	.26-.33	.67	.10	.05	**.85**	.29	.18	.85	.29	.18	-	-	-	.54-.77	.08-.14
LogMapBio	-	-	**.70**	.10	.05	-	-	-	-	-	-	-	-	-	.50-.73	.06-.08
LogMapKG	.56-.96	.26-.33	.67	.10	.05	**.85**	.29	.18	.85	.29	.18	-	-	-	.54-.77	.08-.11
LogMapLt	.50-.87	.23-.31	.66	.10	**.06**	.69	**.36**	**.25**	.69	.36	.25	-	-	-	.25-.35	.08-.11
Wiktionary	.49-.88	.26-.35	-	-	-	-	-	-	-	-	-	-	-	-	**.89-.96**	.08-.11

Three systems were able to generate complex correspondences: AMLC, AROA, and CANARD. The results for the other systems are reported in terms of simple alignments. The results of the systems on the five test cases are summarized in Table 20.

With respect to the Hydrography test cases, only AMLC can generate two correct complex correspondences which are stating that a class in the source ontology is equivalent to the union of two classes in the target ontology. Most of the systems achieved fair results in terms of precision, but the low recall reflects that the current ontology alignment systems still need to be improved to find more complex relations.

In terms of Geolink and populated GeoLink test cases, the real-world instance data from GeoLink Project is also populated into the ontology in order to enable the systems

that depend on instance-based matching algorithms to evaluate their performance. There are three alignment systems that generate complex alignments in GeoLink Benchmark, which are AMLC, AROA, and CANARD. AMLC didn't find any correct complex alignment, while AROA and CANARD achieved relatively good performance. One of the reasons may be that these two systems are instance-based systems, which rely on the shared instances between ontologies. In other words, the shared instance data between two ontologies would be helpful to the matching process.

In the populated Enslaved test case, only AMLC, AROA, and CANARD can produce complex alignments. The relaxed precision of AMLC and AROA look relatively fair, while CANARD reports a lower relaxed precision. AROA found the largest number of the complex correspondences among three systems, while the AMLC outputs the largest number of the simple correspondences.

With respect to the Conference test cases the track has the same participant, AMLC, as the last year. Based on the evaluation the alignments from AMLC now conforms to the EDOAL syntax but otherwise the content of the alignment is the same.

In the Populated Conference test case, AMLC's results precision and coverage scores are lower than last year, probably because it did not take a simple reference alignment as input. CANARD's results are close to last year's. ALIN obtains the best precision score.

In the Taxon dataset, CANARD obtains the best coverage score but its precision has decreased significantly. This year, AMLC could be evaluated on this dataset ; however, the output correspondences did not cover the evaluation queries. The simple matcher obtains approximatively the same coverage score.

A more detailed discussion of the results of each task can be found in the OAEI page for this track. For a third edition of complex matching in an OAEI campaign, and given the inherent difficulty of the task, the results and participation are promising albeit still modest.

5 Conclusions and Lessons Learned

In 2020, we witnessed a slight decrease in the number of participants in comparison with previous years, but with a healthy mix of new and returning systems. However, like last year, the distribution of participants by tracks was uneven. In future editions we should facilitate the participation of non-Java systems (the use of the MELT framework [36] was a step forward this year) and Machine Learning based system by providing partial alignment sets for supervised learning. Furthermore, new systems might use deep learning technology which requires specific hardware like GPUs and the like. An option would be a simple HTTP interface to allow the deployment and evaluation on different machines. The MELT framework can be easily extended with such an interface while at the same time compatibility with SEALS and HOBBIT can be retained.

The **schema matching tracks** saw abundant participation, but, as has been the trend of the recent years, little substantial progress in terms of quality of the results or run time of top matching systems, judging from the long-standing tracks. On the one hand, this may be a sign of a performance plateau being reached by existing strategies and algorithms, which would suggest that new technology is needed to obtain significant

improvements. On the other hand, it is also true that established matching systems tend to focus more on new tracks and datasets than on improving their performance in long-standing tracks, whereas new systems typically struggle to compete with established ones.

The number of matching systems capable of handling very large ontologies has increased slightly over the last years, but is still relatively modest, judging from the **Large Biomedical Ontologies** track. We will aim at facilitating participation in future editions of this track by providing techniques to divide the matching tasks in manageable subtasks (e.g., [40]).

According to the **Conference** track there is still need for an improvement with regard to the ability of matching systems to match properties. To assist system developers in tackling this aspect we provided a more detailed evaluation in terms of the analysis of the false positives per matching system (available on the Conference track web page). This year this has been extended by the inspection of the explanation of the correspondences provided by the systems. As already pointed out last year, less encouraging is the low number of systems concerned with the logical coherence of the alignments they produce, an aspect which is critical for several semantic web applications. Perhaps a more direct approach is needed to promote this topic, such as providing a more in-depth analysis of the causes of incoherence in the evaluation or even organizing a future track focusing on logical coherence alone. It is, however, clear that this is not an easy task. When naively computing coherent alignments correct correspondences may be removed and incorrect ones are kept, and therefore a domain expert should be involved in the validation of different logical solutions [57, 49]. Finally, this year it was shown that matching domain ontology to cross-domain ontology is difficult task for general matching systems. While this has been done as an experiment without announcing beforehand, we suppose to announce this as new test cases within the track for next year.

With respect to the cross-lingual version of Conference, the **MultiFarm** track still attracts a few number of participants implementing specific strategies to deal with ontologies having a terminological layer in different natural languages. Despite this fact, this year new participants came with alternative strategies (i.e, deep learning) with respect to the last campaigns.

The consensus-based evaluation in the **Disease and Phenotype** track offers limited insights into performance, as several matching systems produce a number of unique correspondences which may or may not be correct. In the absence of a true reference alignment, future evaluation should seek to determine whether the unique correspondences contain indicators of correctness, such as semantic similarity, or appear to be noise. Comparison of the task results with embedded mappings of equivalence in the MONDO disease ontology can also be investigated in future evaluation [55].

Despite the quite promising results obtained by matching systems for the **Biodiversity and Ecology track**, the most important observation is that none of the systems has been able to detect mappings established by domain experts. Detecting such correspondences requires the use of domain-specific core knowledge that captures biodiversity concepts. In addition this year, we put the light on the quasi total incapacity of systems to handle SKOS as input format for semantic resources to align.

The **interactive matching track** also witnessed a small number of participants. Three systems participated this year. This is puzzling considering that this track is based on the *Anatomy* and *Conference* test cases, and those tracks had 13 participants. The process of programmatically querying the Oracle class used to simulate user interactions is simple enough that it should not be a deterrent for participation, but perhaps we should look at facilitating the process further in future OAEI editions by providing implementation examples.

The **complex matching track** opens new perspectives in the field of ontology matching. Tackling complex matching automatically is extremely challenging, likely requiring profound adaptations from matching systems, so the fact that there were three participants that were able to generate complex correspondences in this track should be seen as a positive sign of progress to the state of the art in ontology matching. This year automatic evaluation has been introduced following an instance-based comparison approach.

In the **instance matching tracks** participation increased this year for SPIMBENCH as systems became more familiar with the HOBBIT platform and had more time to do the migration. Regarding Spatial benchmark, the systems didn't have newer versions and the number of participants remained the same. Thus, the benchmark and the systems were the exact same as last year. Participation might increase next year as the systems are still updating their versions and new systems are under development. Automatic instance-matching benchmark generation algorithms have been gaining popularity, as evidenced by the fact that they are used in all three instance matching tracks of this OAEI edition. One aspect that has not been addressed in such algorithms is that, if the transformation is too extreme, the correspondence may be unrealistic and impossible to detect even by humans. As such, we argue that *human-in-the-loop* techniques can be exploited to do a preventive quality-checking of generated correspondences, and refine the set of correspondences included in the final reference alignment.

In the **knowledge graph track**, more matchers are able to match rdf:Properties and are thus better suited for real matching cases. In the third year of this track we saw a small improvement in instance alignments but the margin to the baselines is still small. In this year two new systems focused on the KG track but could not submit their systems in time. We thus expect more systems in the upcoming year.

Like in previous OAEI editions, most participants provided a description of their systems and their experience in the evaluation, in the form of OAEI system papers. These papers, like the present one, have not been peer reviewed. However, they are full contributions to this evaluation exercise, reflecting the effort and insight of matching systems developers, and providing details about those systems and the algorithms they implement.

As each year, fruitful discussions at the Ontology Matching point out different directions for future improvements in OAEI. In particular, in terms of new use cases, one potential new track involves matching ontologies of units of measure (OM and QUDT) [51], in order to improve the ability of a digital twin platform to harmonise, integrate and process quantity values. Another track to be included in the next campaign is about the chemical/biological laboratory domain with strong interest from pharmaceutical companies [30, 32].

The Ontology Alignment Evaluation Initiative will strive to remain a reference to the ontology matching community by improving both the test cases and the testing methodology to better reflect actual needs, as well as to promote progress in this field. More information can be found at: `http://oaei.ontologymatching.org`.

Acknowledgements

We warmly thank the participants of this campaign. We know that they have worked hard to have their matching tools executable in time and they provided useful reports on their experience. The best way to learn about the results remains to read the papers that follow.

We are also grateful to Martin Ringwald and Terry Hayamizu for providing the reference alignment for the anatomy ontologies and thank Elena Beisswanger for her thorough support on improving the quality of the dataset.

We thank Andrea Turbati and the AGROVOC team for their very appreciated help with the preparation of the AGROVOC subset ontology. We are also grateful to Catherine Roussey and Nathalie Hernandez for their help on the Taxon alignment.

We also thank for their support the past members of the Ontology Alignment Evaluation Initiative steering committee: Jérôme Euzenat (INRIA, FR), Yannis Kalfoglou (Ricoh laboratories, UK), Miklos Nagy (The Open University,UK), Natasha Noy (Google Inc., USA), Yuzhong Qu (Southeast University, CN), York Sure (Leibniz Gemeinschaft, DE), Jie Tang (Tsinghua University, CN), Heiner Stuckenschmidt (Mannheim Universität, DE), George Vouros (University of the Aegean, GR).

Daniel Faria was supported by the EC H2020 grant 676559 ELIXIR-EXCELERATE and the Portuguese FCT Grant 22231 BioData.pt, co-financed by FEDER.

Ernesto Jimenez-Ruiz has been partially supported by the SIRIUS Centre for Scalable Data Access (Research Council of Norway, project no.: 237889) and the AIDA project (Alan Turing Institute).

Catia Pesquita was supported by the FCT through the LASIGE Strategic Project (UID/CEC/00408/2013) and the research grant PTDC/EEI-ESS/4633/2014.

Irini Fundulaki and Tzanina Saveta were supported by the EU's Horizon 2020 research and innovation programme under grant agreement No 688227 (Hobbit).

Jana Vataščinová and Ondřej Zamazal were supported by the CSF grant no. 18-23964S.

Patrick Lambrix, Huanyu Li, Mina Abd Nikooie Pour and Ying Li have been supported by the Swedish e-Science Research Centre (SeRC), the Swedish Research Council (Vetenskapsrådet, dnr 2018-04147) and the Swedish National Graduate School in Computer Science (CUGS).

Lu Zhou and Pascal Hitzler have been supported by the National Science Foundation under Grant No. 2033521, KnowWhereGraph: Enriching and Linking Cross-Domain Knowledge Graphs using Spatially-Explicit AI Technologies and the Andrew W. Mellon Foundation through the Enslaved project (identifiers 1708-04732 and 1902-06575).

Beyza Yaman has been supported by the European Union's Horizon 2020 research and innovation programme under Marie Sklodowska-Curie grant agreement No. 801522, by Science Foundation Ireland and co-funded by the European Regional Development Fund through the ADAPT Centre for Digital Content Technology [grant number 13/RC/2106] and Ordnance Survey Ireland.

The Biodiversity and Ecology track has been partially funded by the German Research Foundation in the context of the GFBio Project (grant No. SE 553/7-1) and the CRC 1076 AquaDiva, the Leitprojekt der Fraunhofer Gesellschaft in the context of the MED2ICIN project (grant No. 600628) and the German Network for Bioinformatics Infrastructure - de.NBI (grant No. 031A539B). In 2020, the track was also supported by the Data to Knowledge in Agronomy and Biodiversity (D2KAB – www.d2kab.org) project that received funding from the French National Research Agency (ANR-18-CE23-0017). We would like to thank FAO AIMS and US NAL as well as the GACS project for providing mappings betweem AGROVOC and NALT. We would like to thank Christian Pichot and the ANAEE France project for providing mappings betweem ANAEETHES and GEMET.

References

1. Manel Achichi, Michelle Cheatham, Zlatan Dragisic, Jérôme Euzenat, Daniel Faria, Alfio Ferrara, Giorgos Flouris, Irini Fundulaki, Ian Harrow, Valentina Ivanova, Ernesto Jiménez-Ruiz, Kristian Kolthoff, Elena Kuss, Patrick Lambrix, Henrik Leopold, Huanyu Li, Christian Meilicke, Majid Mohammadi, Stefano Montanelli, Catia Pesquita, Tzanina Saveta, Pavel Shvaiko, Andrea Splendiani, Heiner Stuckenschmidt, Élodie Thiéblin, Konstantin Todorov, Cássia Trojahn, and Ondrej Zamazal. Results of the ontology alignment evaluation initiative 2017. In *Proceedings of the 12th International Workshop on Ontology Matching, Vienna, Austria*, pages 61–113, 2017.
2. Manel Achichi, Michelle Cheatham, Zlatan Dragisic, Jerome Euzenat, Daniel Faria, Alfio Ferrara, Giorgos Flouris, Irini Fundulaki, Ian Harrow, Valentina Ivanova, Ernesto Jiménez-Ruiz, Elena Kuss, Patrick Lambrix, Henrik Leopold, Huanyu Li, Christian Meilicke, Stefano Montanelli, Catia Pesquita, Tzanina Saveta, Pavel Shvaiko, Andrea Splendiani, Heiner Stuckenschmidt, Konstantin Todorov, Cássia Trojahn, and Ondrej Zamazal. Results of the ontology alignment evaluation initiative 2016. In *Proceedings of the 11th International Ontology matching workshop, Kobe (JP)*, pages 73–129, 2016.
3. José Luis Aguirre, Bernardo Cuenca Grau, Kai Eckert, Jérôme Euzenat, Alfio Ferrara, Robert Willem van Hague, Laura Hollink, Ernesto Jiménez-Ruiz, Christian Meilicke, Andriy Nikolov, Dominique Ritze, François Scharffe, Pavel Shvaiko, Ondrej Sváb-Zamazal, Cássia Trojahn, and Benjamin Zapilko. Results of the ontology alignment evaluation initiative 2012. In *Proceedings of the 7th International Ontology matching workshop, Boston (MA, US)*, pages 73–115, 2012.
4. Alsayed Algergawy, Michelle Cheatham, Daniel Faria, Alfio Ferrara, Irini Fundulaki, Ian Harrow, Sven Hertling, Ernesto Jiménez-Ruiz, Naouel Karam, Abderrahmane Khiat, Patrick Lambrix, Huanyu Li, Stefano Montanelli, Heiko Paulheim, Catia Pesquita, Tzanina Saveta, Daniela Schmidt, Pavel Shvaiko, Andrea Splendiani, Élodie Thiéblin, Cássia Trojahn, Jana Vatascinová, Ondrej Zamazal, and Lu Zhou. Results of the ontology alignment evaluation initiative 2018. In *Proceedings of the 13th International Workshop on Ontology Matching, Monterey (CA, US)*, pages 76–116, 2018.

5. Alsayed Algergawy, Daniel Faria, Alfio Ferrara, Irini Fundulaki, Ian Harrow, Sven Hertling, Ernesto Jiménez-Ruiz, Naouel Karam, Abderrahmane Khiat, Patrick Lambrix, Huanyu Li, Stefano Montanelli, Heiko Paulheim, Catia Pesquita, Tzanina Saveta, Pavel Shvaiko, Andrea Splendiani, Élodie Thiéblin, Cássia Trojahn, Jana Vatascinová, Ondrej Zamazal, and Lu Zhou. Results of the ontology alignment evaluation initiative 2019. In *Proceedings of the 14th International Workshop on Ontology Matching, Auckland, New Zealand*, pages 46–85, 2019.
6. R Amini, L Zhou, and P Hitzler. Geolink cruises: A non-synthetic benchmark for coreference resolution on knowledge graphs. In *29th ACM International Conference on Information and Knowledge Management*, 2020.
7. Benhamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors. *Proc. K-Cap Workshop on Integrating Ontologies*, Banff (Canada), 2005.
8. Olivier Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32:267–270, 2004.
9. Pier Luigi Buttigieg, Norman Morrison, Barry Smith, Christopher J. Mungall, and Suzanna E. Lewis. The environment ontology: contextualising biological and biomedical entities. *Biomedical Semantics*, 4(1):43, December 2013.
10. Caterina Caracciolo, Jérôme Euzenat, Laura Hollink, Ryutaro Ichise, Antoine Isaac, Véronique Malaisé, Christian Meilicke, Juan Pane, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Vojtech Svátek. Results of the ontology alignment evaluation initiative 2008. In *Proceedings of the 3rd Ontology matching workshop, Karlsruhe (DE)*, pages 73–120, 2008.
11. Michelle Cheatham, Zlatan Dragisic, Jérôme Euzenat, Daniel Faria, Alfio Ferrara, Giorgos Flouris, Irini Fundulaki, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Patrick Lambrix, Stefano Montanelli, Catia Pesquita, Tzanina Saveta, Pavel Shvaiko, Alessandro Solimando, Cássia Trojahn, and Ondřej Zamazal. Results of the ontology alignment evaluation initiative 2015. In *Proceedings of the 10th International Ontology matching workshop, Bethlehem (PA, US)*, pages 60–115, 2015.
12. Michelle Cheatham, Dalia Varanka, Fatima Arauz, and Lu Zhou. Alignment of surface water ontologies: a comparison of manual and automated approaches. *J. Geogr. Syst.*, 22(2):267–289, 2020.
13. Jean Clobert, André Chanzy, Jean-François Le Galliard, Abad Chabbi, Lucile Greiveldinger, Thierry Caquet, Michel Loreau, Christian Mougin, Christian Pichot, Jacques Roy, et al. How to integrate experimental research approaches in ecological and environmental studies: Anaee france as an example. *Frontiers in Ecology and Evolution*, 6:43, 2018.
14. Laurel Cooper, Ramona L. Walls, Justin Elser, Maria A. Gandolfo, Dennis W. Stevenson, Barry Smith, Justin Preece, Balaji Athreya, Christopher J. Mungall, Stefan Rensing, Manuel Hiss, Daniel Lang, Ralf Reski, Tanya Z. Berardini, Donghui Li, Eva Huala, Mary Schaeffer, Naama Menda, Elizabeth Arnaud, Rosemary Shrestha, Yukiko Yamazaki, and Pankaj Jaiswal. The Plant Ontology as a Tool for Comparative Plant Anatomy and Genomic Analyses. *Plant and Cell Physiology*, 54(2):e1, December 2012.
15. Bernardo Cuenca Grau, Zlatan Dragisic, Kai Eckert, Jérôme Euzenat, Alfio Ferrara, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Andreas Oskar Kempf, Patrick Lambrix, Andriy Nikolov, Heiko Paulheim, Dominique Ritze, François Scharffe, Pavel Shvaiko, Cássia Trojahn dos Santos, and Ondrej Zamazal. Results of the ontology alignment evaluation initiative 2013. In Pavel Shvaiko, Jérôme Euzenat, Kavitha Srinivas, Ming Mao, and Ernesto Jiménez-Ruiz, editors, *Proceedings of the 8th International Ontology matching workshop, Sydney (NSW, AU)*, pages 61–100, 2013.
16. Jim Dabrowski and Ethan V. Munson. 40 years of searching for the best computer system response time. *Interacting with Computers*, 23(5):555–564, 2011.

17. Thaleia Dimitra Doudali, Ioannis Konstantinou, and Nectarios Koziris Doudali. Spaten: a Spatio-Temporal and Textual Big Data Generator. In *IEEE Big Data*, pages 3416–3421, 2017.
18. Zlatan Dragisic, Kai Eckert, Jérôme Euzenat, Daniel Faria, Alfio Ferrara, Roger Granada, Valentina Ivanova, Ernesto Jiménez-Ruiz, Andreas Oskar Kempf, Patrick Lambrix, Stefano Montanelli, Heiko Paulheim, Dominique Ritze, Pavel Shvaiko, Alessandro Solimando, Cássia Trojahn dos Santos, Ondrej Zamazal, and Bernardo Cuenca Grau. Results of the ontology alignment evaluation initiative 2014. In *Proceedings of the 9th International Ontology matching workshop, Riva del Garda (IT)*, pages 61–104, 2014.
19. Zlatan Dragisic, Valentina Ivanova, Patrick Lambrix, Daniel Faria, Ernesto Jiménez-Ruiz, and Catia Pesquita. User validation in ontology alignment. In *Proceedings of the 15th International Semantic Web Conference, Kobe (JP)*, pages 200–217, 2016.
20. Zlatan Dragisic, Valentina Ivanova, Huanyu Li, and Patrick Lambrix. Experiences from the anatomy track in the ontology alignment evaluation initiative. *Journal of Biomedical Semantics*, 8:56:1–56:28, 2017.
21. Marc Ehrig and Jérôme Euzenat. Relaxed precision and recall for ontology matching. In *Integrating Ontologies, Proceedings of the K-CAP Workshop on Integrating Ontologies, Banff, Canada*, 2005.
22. Jérôme Euzenat, Alfio Ferrara, Laura Hollink, Antoine Isaac, Cliff Joslyn, Véronique Malaisé, Christian Meilicke, Andriy Nikolov, Juan Pane, Marta Sabou, François Scharffe, Pavel Shvaiko, Vassilis Spiliopoulos, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Cássia Trojahn dos Santos, George Vouros, and Shenghui Wang. Results of the ontology alignment evaluation initiative 2009. In *Proceedings of the 4th International Ontology matching workshop, Chantilly (VA, US)*, pages 73–126, 2009.
23. Jérôme Euzenat, Alfio Ferrara, Christian Meilicke, Andriy Nikolov, Juan Pane, François Scharffe, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, and Cássia Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2010. In *Proceedings of the 5th International Ontology matching workshop, Shanghai (CN)*, pages 85–117, 2010.
24. Jérôme Euzenat, Alfio Ferrara, Robert Willem van Hague, Laura Hollink, Christian Meilicke, Andriy Nikolov, François Scharffe, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Cássia Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2011. In *Proceedings of the 6th International Ontology matching workshop, Bonn (DE)*, pages 85–110, 2011.
25. Jérôme Euzenat, Antoine Isaac, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2007. In *Proceedings 2nd International Ontology matching workshop, Busan (KR)*, pages 96–132, 2007.
26. Jérôme Euzenat, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, and Cássia Trojahn dos Santos. Ontology alignment evaluation initiative: six years of experience. *Journal on Data Semantics*, XV:158–192, 2011.
27. Jérôme Euzenat, Malgorzata Mochol, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2006. In *Proceedings of the 1st International Ontology matching workshop, Athens (GA, US)*, pages 73–95, 2006.
28. Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, 2nd edition, 2013.
29. Daniel Faria, Ernesto Jiménez-Ruiz, Catia Pesquita, Emanuel Santos, and Francisco M. Couto. Towards Annotating Potential Incoherences in BioPortal Mappings. In *Proceedings of the 13th International Semantic Web Conference*, volume 8797, pages 17–32, 2014.
30. I. Harrow et al. Ontology mapping for semantically enabled applications. *Drug Discovery Today*, 2019.

31. Ian Harrow, Ernesto Jiménez-Ruiz, Andrea Splendiani, Martin Romacker, Peter Woollard, Scott Markel, Yasmin Alam-Faruque, Martin Koch, James Malone, and Arild Waaler. Matching Disease and Phenotype Ontologies in the Ontology Alignment Evaluation Initiative. *Journal of Biomedical Semantics*, 8:55:1–55:13, 2017.
32. Ian Harrow, Thomas Liener, and Ernesto Jiménez-Ruiz. Ontology matching for the laboratory analytics domain. In *Proceedings of the 15th International Workshop on Ontology Matching*, 2020.
33. Sven Hertling and Heiko Paulheim. Dbkwik: A consolidated knowledge graph from thousands of wikis. In *Proceedings of the International Conference on Big Knowledge*, 2018.
34. Sven Hertling and Heiko Paulheim. Dbkwik: extracting and integrating knowledge from thousands of wikis. *Knowledge and Information Systems*, 2019.
35. Sven Hertling and Heiko Paulheim. The knowledge graph track at oaei - gold standards, baselines, and the golden hammer bias. In *The Semantic Web: ESWC 2020*, pages 343–359, 2020.
36. Sven Hertling, Jan Portisch, and Heiko Paulheim. Melt - matching evaluation toolkit. In Maribel Acosta, Philippe Cudré-Mauroux, Maria Maleshkova, Tassilo Pellegrini, Harald Sack, and York Sure-Vetter, editors, *Semantic Systems. The Power of AI and Knowledge Graphs*, pages 231–245, Cham, 2019. Springer International Publishing.
37. Sven Hertling, Jan Portisch, and Heiko Paulheim. Melt - matching evaluation toolkit. In *SEMANTICS*, 2019.
38. Robert Hoehndorf, Mona Alshahrani, Georgios V Gkoutos, George Gosline, Quentin Groom, Thomas Hamann, Jens Kattge, Sylvia Mota de Oliveira, Marco Schmidt, Soraya Sierra, et al. The flora phenotype ontology (flopo): tool for integrating morphological traits and phenotypes of vascular plants. *Journal of Biomedical Semantics*, 7(1):1–11, 2016.
39. Valentina Ivanova, Patrick Lambrix, and Johan Åberg. Requirements for and evaluation of user support for large-scale ontology alignment. In *Proceedings of the European Semantic Web Conference*, pages 3–20, 2015.
40. Ernesto Jiménez-Ruiz, Asan Agibetov, Jiaoyan Chen, Matthias Samwald, and Valerie Cross. Dividing the Ontology Alignment Task with Semantic Embeddings and Logic-Based Modules. In *24th European Conference on Artificial Intelligence (ECAI)*, pages 784–791, 2020.
41. Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. LogMap: Logic-based and scalable ontology matching. In *Proceedings of the 10th International Semantic Web Conference, Bonn (DE)*, pages 273–288, 2011.
42. Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga. Logic-based assessment of the compatibility of UMLS ontology sources. *J. Biomed. Sem.*, 2, 2011.
43. Ernesto Jiménez-Ruiz, Christian Meilicke, Bernardo Cuenca Grau, and Ian Horrocks. Evaluating mapping repair systems with large biomedical ontologies. In *Proceedings of the 26th Description Logics Workshop*, 2013.
44. Ernesto Jiménez-Ruiz, Tzanina Saveta, Ondvrej Zamazal, Sven Hertling, Michael Röder, Irini Fundulaki, Axel-Cyrille Ngonga Ngomo, Mohamed Ahmed Sherif, Amina Annane, Zohra Bellahsene, Sadok Ben Yahia, Gayo Diallo, Daniel Faria, Marouen Kachroudi, Abderrahmane Khiat, Patrick Lambrix, Huanyu Li, Maximilian Mackeprang, Majid Mohammadi, Maciej Rybinski, Booma Sowkarthiga Balasubramani, and Cassia Trojahn. Introducing the HOBBIT platform into the Ontology Alignment Evaluation Campaign. In *Proceedings of the 13th International Workshop on Ontology Matching*, 2018.
45. Naouel Karam, Abderrahmane Khiat, Alsayed Algergawy, Melanie Sattler, Claus Weiland, and Marco Schmidt. Matching biodiversity and ecology ontologies: challenges and evaluation results. *Knowl. Eng. Rev.*, 35:e9, 2020.
46. Naouel Karam, Claudia Müller-Birn, Maren Gleisberg, David Fichtmüller, Robert Tolksdorf, and Anton Güntsch. A terminology service supporting semantic annotation, integration,

discovery and analysis of interdisciplinary research data. *Datenbank-Spektrum*, 16(3):195–205, 2016.

47. Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. Concurrent classification of EL ontologies. In *Proceedings of the 10th International Semantic Web Conference, Bonn (DE)*, pages 305–320, 2011.
48. Friederike Klan, Erik Faessler, Alsayed Algergawy, Birgitta König-Ries, and Udo Hahn. Integrated semantic search on structured and unstructured data in the adonis system. In *Proceedings of the 2nd International Workshop on Semantics for Biodiversity*, 2017.
49. Patrick Lambrix. Completing and debugging ontologies: state of the art and challenges. *CoRR*, abs/1908.03171, 2019.
50. Huanyu Li, Zlatan Dragisic, Daniel Faria, Valentina Ivanova, Ernesto Jiménez-Ruiz, Patrick Lambrix, and Catia Pesquita. User validation in ontology alignment: functional assessment and impact. *The Knowledge Engineering Review*, 34:e15, 2019.
51. Francisco Martín-Recuerda, Dirk Walther, Siegfried Eisinger, Graham Moore, Petter Andersen, Per-Olav Opdahl, and Lillian Hella. Revisiting ontologies of units of measure for harmonising quantity values - A use case. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part II*, volume 12507 of *Lecture Notes in Computer Science*, pages 551–567. Springer, 2020.
52. Christian Meilicke. *Alignment Incoherence in Ontology Matching*. PhD thesis, University Mannheim, 2011.
53. Christian Meilicke, Raúl García Castro, Frederico Freitas, Willem Robert van Hage, Elena Montiel-Ponsoda, Ryan Ribeiro de Azevedo, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Andrei Tamilin, Cássia Trojahn, and Shenghui Wang. MultiFarm: A benchmark for multilingual ontology matching. *Journal of web semantics*, 15(3):62–68, 2012.
54. Boris Motik, Rob Shearer, and Ian Horrocks. Hypertableau reasoning for description logics. *Journal of Artificial Intelligence Research*, 36:165–228, 2009.
55. Christopher J Mungall, Julie A McMurry, Sebastian Köhler, James P. Balhoff, Charles Borromeo, Matthew Brush, Seth Carbon, Tom Conlin, Nathan Dunn, Mark Engelstad, Erin Foster, JP Gourdine, Julius O.B. Jacobsen, Daniel Keith, Bryan Laraway, Suzanna E. Lewis, Jeremy Nguyen Xuan, Kent Shefchek, Nicole Vasilevsky, Zhou Yuan, Nicole Washington, Harry Hochheiser, Tudor Groza, Damian Smedley, Peter N. Robinson, and Melissa A Haendel. The Monarch Initiative: An integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Res.*, 45, 2017.
56. Heiko Paulheim, Sven Hertling, and Dominique Ritze. Towards evaluating interactive ontology matching tools. In *Proceedings of the 10th Extended Semantic Web Conference, Montpellier (FR)*, pages 31–45, 2013.
57. Catia Pesquita, Daniel Faria, Emanuel Santos, and Francisco M. Couto. To repair or not to repair: reconciling correctness and coherence in ontology reference alignments. In Pavel Shvaiko, Jérôme Euzenat, Kavitha Srinivas, Ming Mao, and Ernesto Jiménez-Ruiz, editors, *Proceedings of the 8th International Workshop on Ontology Matching co-located with the 12th International Semantic Web Conference (ISWC 2013), Sydney, Australia, October 21, 2013*, volume 1111 of *CEUR Workshop Proceedings*, pages 13–24. CEUR-WS.org, 2013.
58. Robert G Raskin and Michael J Pan. Knowledge representation in the semantic web for earth and environmental terminology (sweet). *Computers & geosciences*, 31(9):1119–1125, 2005.
59. Johannes Keizer Sachit Rajbhandari. The AGROVOC Concept Scheme ; A Walkthrough. *Integrative Agriculture*, 11(5):694–699, May 2012.
60. Emanuel Santos, Daniel Faria, Catia Pesquita, and Francisco M Couto. Ontology alignment repair through modularization and confidence-based heuristics. *PLoS ONE*, 10(12):e0144807, 2015.

61. Martin Šatra and Ondrej Zamazal. Towards matching of domain ontologies to cross-domain ontology: Evaluation perspective. In *Proceedings of the 19th International Workshop on Ontology Matching*, 2020.
62. Tzanina Saveta, Evangelia Daskalaki, Giorgos Flouris, Irini Fundulaki, Melanie Herschel, and Axel-Cyrille Ngonga Ngomo. Pushing the limits of instance matching systems: A semantics-aware benchmark for linked data. In *Proceedings of the 24th International Conference on World Wide Web*, pages 105–106, New York, NY, USA, 2015. ACM.
63. Alessandro Solimando, Ernesto Jiménez-Ruiz, and Giovanna Guerrini. Detecting and correcting conservativity principle violations in ontology-to-ontology mappings. In *Proceedings of the International Semantic Web Conference*, pages 1–16. Springer, 2014.
64. Alessandro Solimando, Ernesto Jimenez-Ruiz, and Giovanna Guerrini. Minimizing conservativity violations in ontology alignments: Algorithms and evaluation. *Knowledge and Information Systems*, 2016.
65. Christian Strobl. *Encyclopedia of GIS*, chapter Dimensionally Extended Nine-Intersection Model (DE-9IM), pages 240–245. Springer, 2008.
66. York Sure, Oscar Corcho, Jérôme Euzenat, and Todd Hughes, editors. *Proceedings of the Workshop on Evaluation of Ontology-based Tools (EON), Hiroshima (JP)*, 2004.
67. Élodie Thiéblin. Do competency questions for alignment help fostering complex correspondences? In *Proceedings of the EKAW Doctoral Consortium 2018*, 2018.
68. Élodie Thiéblin, Fabien Amarger, Ollivier Haemmerlé, Nathalie Hernandez, and Cássia Trojahn dos Santos. Rewriting SELECT SPARQL queries from 1: n complex correspondences. In *Proceedings of the 11th International Workshop on Ontology Matching*, pages 49–60, 2016.
69. Elodie Thiéblin, Michelle Cheatham, Cassia Trojahn, Ondrej Zamazal, and Lu Zhou. The First Version of the OAEI Complex Alignment Benchmark. In *Proceedings of the International Semantic Web Conference (Posters and Demos)*, 2018.
70. Ondřej Zamazal and Vojtěch Svátek. The ten-year ontofarm and its fertilization within the onto-sphere. *Web Semantics: Science, Services and Agents on the World Wide Web*, 43:46–53, 2017.
71. L Zhou, C Shimizu, P Hitzler, A Sheill, S Estrecha, C Foley, D Tarr, and Rehberger D. The enslaved dataset: A real-world complex ontology alignment benchmark using wikibase. In *29th ACM International Conference on Information and Knowledge Management*, 2020.
72. Lu Zhou, Michelle Cheatham, Adila Krisnadhi, and Pascal Hitzler. A complex alignment benchmark: Geolink dataset. In *Proceedings of the 17th International Semantic Web Conference, Monterey (CA, USA)*, pages 273–288, 2018.
73. Lu Zhou, Michelle Cheatham, Adila Krisnadhi, and Pascal Hitzler. Geolink data set: A complex alignment benchmark from real-world ontology. *Data Intell.*, 2(3):353–378, 2020.

Linköping, Jena, Lisboa, Heraklion, Mannheim, Montpellier, Oslo, London, Berlin,
Sankt Augustin, Trento, Toulouse, Prague, Manhattan, Dublin
December 2020

ALIN Results for OAEI 2020

Jomar da Silva[1], Carla Delgado[1],
Kate Revoredo[2], and Fernanda Araujo Baião[3]

[1] Graduate Program in Informatics
Federal University of Rio de Janeiro (UFRJ), Brazil
[2] Vienna University of Economics and Business, Vienna, Austria
[3] Department of Industrial Engineering
Pontifical Catholic University of Rio de Janeiro (PUC-Rio), Brazil
jomar.silva@uniriotec.br,carla@ppgi.ufrj.br,
kate.revoredo@wu.ac.at,fbaiao@puc-rio.br

Abstract. [4]
ALIN is a system for interactive ontology matching. The ALIN version participating in OAEI 2020 applies natural language processing techniques (NLP) to standardize the concept names of the ontologies that participate in the matching process. As ALIN selects through semantic and lexical metrics many of the mappings that the domain expert evaluates, we hope that the standardization of the concept names will improve the selection of the mappings and thus the generated alignment. This article describes the participation of ALIN at OAEI 2020 and discusses its results.

Keywords: ontology matching, Wordnet, interactive ontology matching, ontology alignment, interactive ontology alignment, natural language processing

1 Presentation of the system

Due to the advances in information and communication technologies, a large amount of data repositories became available. Those repositories, however, are highly semantically heterogeneous, which hinders their integration. Ontology Matching has been successfully applied to solve this problem, by discovering mappings between two distinct ontologies which, in turn, conceptually define the data stored in each repository. The Ontology Matching process seeks to discover correspondences (mappings) between entities of different ontologies, and this may be performed manually, semi-automatically or automatically [1]. Among all semi-automatic approaches, the ones that follow an interactive strategy stand out, considering the knowledge of domain experts through their participation during the matching process [2]. The use of a domain expert is not always possible since it is an expensive, scarce and time-consuming resource; when available,

however, this strategy has achieved results that are superior to automatic (non-interactive) strategies. Nevertheless, there is still room for improvements [2], as evidenced by the most recent results from the evaluation of interactive tools in the OAEI[5] (Ontology Alignment Evaluation Initiative). ALIN [3] is a system for interactive ontology matching which has been participating in all OAEI editions since 2016, with increasingly improved results.

1.1 State, Purpose and General statement

Interactive ontology matching systems select mappings for domain expert evaluates. ALIN selects many of these mappings through semantic and lexical metrics. As the concept names of the ontologies are not standardized, these metrics may return lower values than would be the case if they were standardized. This smaller metric may cause ALIN not to select these mappings for evaluation by the domain expert. In its 2020 version, ALIN proposes Natural Language Processing (NLP) techniques such as the development of regular grammars (in reality its equivalent regular expressions) and context free grammars along with their respective lexical analyzers (scanners) and syntax analyzers (parsers), for the concept names of the ontologies to be matched. The use of these NLP resources (scanners and parsers) makes it possible to translate different patterns used in the two ontologies into a unique one. This standardization allows ALIN to select better mappings for the domain expert to evaluate.

To do the standardization, ALIN will have a new phase before the execution of the program. In this phase, an NLP expert develops, manually, grammars to the concept names of the ontologies and their respective scanners and parsers. ALIN uses these scanners and parsers during the execution of the program. This new phase is possible in an interactive ontology matching system because:

1. We know before the program runs which ontologies it will match, as we need to look for experts in the domain of ontologies to interact with the program;
2. The process of searching, meeting, and scheduling a day available for the expert to participate in the process can take a long time, probably a few days.

We can use this time of a few days until the execution of the program to develop the necessary grammars, scanners, and parsers for the ontologies. In this version of ALIN, the authors of this paper played the role of the NLP expert.

1.2 Specific techniques used

During its matching process, ALIN handles three sets of mappings: (i) Accepted, which is a set of mappings definitely to be retained in the alignment; (ii) Selected, which is a set of mappings where each is yet to be decided if it will be included in the alignment; and (iii) Suspended, which is a set of mappings that have

[5] Available at http://oaei.ontologymatching.org/2020/results/interactive/index.html, last accessed on Oct, 23, 2020.

been previously selected, but (temporarily or permanently) filtered out of the alignment.

Given the previous definitions, ALIN procedure follows 5 Steps, described as follows:

1. Select mappings: select the first mappings and automatically accepts some of them. We explain the selection and acceptance process below;
2. Filter mappings: suspend some selected mappings, using lexical criteria for that;
3. Ask domain expert: accepts or rejects selected mappings, according to domain expert feedback
4. Propagate: select new mappings, reject some selected mappings or unsuspend some suspended mappings (depending on newly accepted mappings)
5. Go back to 3 as long as there are undecided selected mappings

All versions of ALIN (since its very first OAEI participation) follow this general procedure. In this 2020 version, ALIN includes a new step where an NLP expert develops grammars, and their respective scanners, and parsers to the concept names of the ontologies. ALIN uses these scanners and parsers to standardize the concept names of the ontologies and thus improve the generated alignment. The new step can lead to, for example, correcting spelling errors and unifying different spellings for the same concept name. More detailed examples of possible standardization of concept names are presented in [4]. ALIN uses the developed scanners and parsers in step 1 of the program.

ALIN applies the following techniques:

- Step 1. ALIN runs the scanners and the parsers for each concept name of the ontologies, modifying it and standardizing it. ALIN uses a blocking strategy where it discards all data properties and object properties of the ontologies. So, in this step, ALIN selects only concept mappings, using linguistic similarities between the concept names. ALIN automatically accepts concept mappings whose names are synonyms. ALIN uses the Wordnet and domain-specific ontologies (the FMA Ontology in the Anatomy track) to find synonyms between entities.
- Step 2. ALIN suspends the selected mappings whose entities have low lexical similarity. We use the Jaccard, Jaro-Wrinkler, and n-gram lexical metrics to calculate the lexical similarity of the selected mappings. We based the process of choosing the similarity metrics used by ALIN on the result of these metrics in assessments [5]. It is relevant to know that these suspended mappings can be further unsuspended later, as proposed in [6].
- Step 3. At this point, the domain expert interaction begins. ALIN sorts the selected mappings in a descending order according to the sum of similarity metric values. The sorted selected mappings are submitted to the domain expert.
- Step 4. Initially, the set of selected mappings contains only concept mappings. At each interaction with the domain expert, if s/he accepts the mapping, ALIN (i) removes from the set of selected mappings all the mappings

that compose an instantiation of a mapping anti-pattern [7][8] (we explain mapping anti-patterns below) with the accepted mappings; (ii) selects data property (like [9]) and object property mappings related to the accepted concept mappings; (iii) unsuspends all concept mappings whose both entities are subconcepts of the concept of an accepted mapping, following a similar technique proposed in our previous work [6].
- Step 5. The interaction phase continues until there are no selected mappings.

There are logical constraints which should apply to several ontologies. For example, an ontology may have construction constraints, such as a concept cannot be equivalent to its superconcept. An alignment may have other constraints like, for example, an entity of ontology O cannot be equivalent to two entities of the ontology O'. A mapping anti-pattern is a combination of mappings that generates a problematic alignment, i.e., a logical inconsistency or a violated constraint.

1.3 Link to the system and parameters file

To this version, ALIN used the scanners and the parsers we developed for the ontologies of the conference and anatomy tracks.

ALIN is available [6] as a package to be run through the SEALS client.

2 Results

Interactive ontology matching is the focus of the ALIN system. If you compare the participation of ALIN in 2020 and 2019 (Table 4), you will see an improvement in the quality of the generated alignment, showing the effectiveness of the techniques used.

2.1 Comments on the participation of ALIN in non-interactive tracks

The use of NLP techniques led to an increase in the F-Measure of non-interactively generated alignments in the Anatomy track but stability on the Conference track (Table 1).

2.2 Comments on the participation of ALIN in interactive tracks

In the Anatomy track, ALIN was better than LogMap in both quality (F-Measure) and total requests, but worse in both aspects than AML (Table 2). In the Conference track, ALIN was first in quality and third in total requests (Table 3).

[6] *https://drive.google.com/file/d/1ZM3g0aOgUha-9VptUbqk9nmnkFCl7L/view?usp=sharing*

Table 1. Participation of ALIN in Anatomy Non-Interactive Track - 2019[10]/2020[11] and Conference Non-Interactive Track - 2019[10]/2020[12]

	Year	Precision	Recall	F-measure
Anatomy track	2019	0.974	0.698	0.813
	2020	0.986	0.72	0.832
	Year	Precision	Recall	F-measure
Conference track	2019	0.82	0.43	0.56
	2020	0.82	0.43	0.56

Table 2. Participation of ALIN in Anatomy Interactive Track - Error Rate 0.0[13]

Tool	Precision	Recall	F-measure	Total Requests
ALIN	0.988	0.856	0.917	360
AML	0.972	0.933	0.952	189
LogMap	0.988	0.846	0.912	388

Table 3. Participation of ALIN in Conference Interactive Track - Error Rate 0.0[13]

Tool	Precision	Recall	F-measure	Total Requests
ALIN	0.915	0.705	0.796	233
AML	0.91	0.698	0.79	221
LogMap	0.886	0.61	0.723	82

Interactive Anatomy Track In this track, ALIN had a decrease in the number of interactions with the domain expert and an increase in the quality of the generated alignment, showing that the use of the NLP techniques are effective for this track (Table 4).

Interactive Conference Track In this track, ALIN had an increase in the quality of the generated alignment but an increase in the number of domain expert interactions (Table 5).

2.3 Comparison of the participation of ALIN in OAEI 2020 with its participation in OAEI 2019

The quality of the alignment generated by ALIN depends on the correct feedback from the domain expert, as ALIN uses this feedback to select new mappings. When ALIN selects wrong mappings, the quality of the generated alignment tends to decrease. If we compare this year's quality decline with last year's, we see that this fall is more sharp (Table 6).

The run time of ALIN this year was shorter than last year (Table 7). In an Intel I5 with 10Gb reserved to ALIN, ALIN has run 20% faster this year than last

year. The execution in OAEI had a reduction in the run time, but other systems also had this reduction. So this difference may be due both to modifications made in ALIN and to changes in the computational environment.

Table 4. Participation of ALIN in Anatomy Interactive Track - OAEI 2016[14]/2017[15]/2018[16]/2019[10]/2020[13] - Error Rate 0.0

Year	Precision	Recall	F-measure	Total Requests
2016	0.993	0.749	0.854	803
2017	0.993	0.794	0.882	939
2018	0.994	0.826	0.902	602
2019	0.979	0.85	0.91	365
2020	0.988	0.856	0.917	360

Table 5. Participation of ALIN in Conference Interactive Track - OAEI 2016[14]/2017[15]/2018[16]/2019[10]/2020[13] - Error Rate 0.0

Year	Precision	Recall	F-measure	Total Requests
2016	0.957	0.735	0.831	326
2017	0.957	0.731	0.829	329
2018	0.921	0.721	0.809	276
2019	0.914	0.695	0.79	228
2020	0.915	0.705	0.796	233

Table 6. F-Measure of ALIN in Anatomy Interactive Track - OAEI /2019[10]/2020[13] and in Conference Interactive Track - OAEI /2019[10]/2020[13] - with Different Error Rates

	Year	Error rate 0.0	Error rate 0.1
Anatomy	2019	0.91	0.889
	2020	0.917	0.887

	Year	Error rate 0.0	Error rate 0.1
Conference	2019	0.79	0.725
	2020	0.796	0.713

Table 7. Run Time (sec) in Anatomy Interactive Track - OAEI /2019[10]/2020[13] and in Conference interactive track - OAEI /2019[10]/2020[13]

	Tool	2019	2020
Anatomy	ALIN	2132	1152
	AML	82	37,3
	LogMap	29	7,6

	Tool	2019	2020
Conference	ALIN	397	136,9
	AML	34	30.1
	LogMap	37	37.96

3 General comments

Evaluating the OAEI 2020 results, ALIN has improved the quality of the generated alignment in the interactive track. However, an increase in the user error rate led to a slight worse alignment. Finally, the number of interactions with the expert was relatively stable since last year, with a slight increase (from 228 to 233 requests) in the Conference track and a slight decrease (from 365 to 360 requests) in the Anatomy track.

Another consideration is that this version of ALIN generates the need for a new expert involved in the process, to develop artifacts (scanner, parser) required for scanning and parsing the name of the concepts. This NLP expert may not always be available, but if he is, the results have shown that his work can improve the quality of the generated alignment.

3.1 Conclusions

ALIN 2020 used NLP techniques to improve the standardization of the concept names of the ontologies to be matched. They have been effective in increasing the quality of the generated alignment while being relatively stable with regard to the number of requests to the user. ALIN had a decrease in run time but a more sharp fall in the alignment quality when the domain expert makes mistakes. An assumption that ALIN now assumes with the inclusion of NLP techniques is the need of a scanner and a parser for the ontologies involved in the matching.

References

1. Euzenat, J., Shvaiko, P.: Ontology Matching - Second Edition. Springer-Verlag (2013)
2. Li, H., Dragisic, Z., Faria, D., Ivanova, V., Jimenez-Ruiz, E., Lambrix, P., Pesquita, C.: User validation in ontology alignment: functional assessment and impact. The Knowledge Engineering Review (2019)

3. Da Silva, J., Revoredo, K., Baião, F., Euzenat, J.: Alin: improving interactive ontology matching by interactively revising mapping suggestions. The Knowledge Engineering Review **35** (2020)
4. Real, F.J.Q., Bella, G., McNeill, F., Bundy, A.: Using domain lexicon and grammar for ontology matching. (2020)
5. Cheatham, M., Hitzler, P.: String similarity metrics for ontology alignment. In: Proceedings of the 12th International Semantic Web Conference - Part II. ISWC '13, New York, NY, USA, Springer-Verlag New York, Inc. (2013) 294–309
6. Silva, J., Baião, F., Revoredo, K., Euzenat, J.: Semantic interactive ontology matching: Synergistic combination of techniques to improve the set of candidate correspondences. In: OM-2017: Proceedings of the Twelfth International Workshop on Ontology Matching. Volume 2032. (2017) 13–24
7. Guedes, A., Baião, F., Shivaprabhu, Revoredo, R.: On the Identification and Representation of Ontology Correspondence Antipatterns. In: Proc. 5th Int. Conf. Ontol. Semant. Web Patterns (WOP'14), CEUR Work. Proc. (2014)
8. Guedes, A., Baião, F., Revoredo, K.: Digging Ontology Correspondence Antipatterns. In: Proceeding WOP'14 Proc. 5th Int. Conf. Ontol. Semant. Web Patterns. Volume 1032. (2014) 38—-48
9. Silva, J., Revoredo, K., Baião, F.A., Euzenat, J.: Interactive Ontology Matching: Using Expert Feedback to Select Attribute Mappings. In: CEUR Workshop Proceedings. Volume 2288. (2018) 25–36
10. Silva, J., Delgado, C., Revoredo, K., Baião, F.: Alin results for oaei 2019. In: Proceedings of the 14th International Workshop on Ontology Matching. OM'19 (2019) 94–100
11. : Results for oaei 2020 - anatomy track. http://oaei.ontologymatching.org/2020/results/anatomy/ Accessed: 2020-10-23.
12. : Results of evaluation for the conference track within oaei 2020. http://oaei.ontologymatching.org/2020/results/conference/index.html Accessed: 2020-10-23.
13. : Results for oaei 2020 - interactive track. http://oaei.ontologymatching.org/2020/results/interactive/index.html Accessed: 2020-10-23.
14. Silva, J., Baião, F., Revoredo, K.: Alin results for oaei 2016. In: OM-2016: Proceedings of the Eleventh International Workshop on Ontology Matching. OM'16 (2016) 130–137
15. Silva, J., Baião, F., Revoredo, K.: Alin results for oaei 2017. In: OM-2017: Proceedings of the Twelfth International Workshop on Ontology Matching. OM'17 (2017) 114–121
16. Silva, J., Baião, F., Revoredo, K.: Alin results for oaei 2018. In: Ontology Matching: OM-2018: Proceedings of the ISWC Workshop. OM'18 (2018) 117–124

ALOD2Vec Matcher Results for OAEI 2020

Jan Portisch[1,2][0000−0001−5420−0663], Michael Hladik[2][0000−0002−2204−3138], and Heiko Paulheim[1][0000−0003−4386−8195]

[1] Data and Web Science Group, University of Mannheim, Germany
{jan, heiko}@informatik.uni-mannheim.de
[2] SAP SE Product Engineering Financial Services, Walldorf, Germany
{jan.portisch, michael.hladik}@sap.com

Abstract. This paper presents the results of the *ALOD2Vec Matcher* in the *Ontology Alignment Evaluation Initiative* (OAEI) 2020. The matching system exploits a Web-scale dataset, i.e. *WebIsALOD*, as background knowledge source. In order to make use of the dataset, the *RDF2Vec* approach is applied to derive embeddings for each concept available in the dataset. *ALOD2Vec Matcher* participated in the OAEI 2018 campaign before. This is the system's second participation. The matching system has been extended, improved, and achieves better results this year.[3]

Keywords: Ontology Matching · Ontology Alignment · External Resources · Background Knowledge · Knowledge Graph Embeddings · RDF2Vec

1 Presentation of the System

1.1 State, Purpose, General Statement

The *ALOD2Vec Matcher* is an element-level, label-based matcher which uses a large-scale Web-crawled RDF dataset of hypernymy relations as general purpose background knowledge. The dataset contains many tail-entities as well as instance data such as persons or places which cannot be found in common thesauri. In order to exploit the external dataset, a neural language model approach is used to obtain a vector for each concept contained in the dataset. This matching system system was initially introduced at the OAEI 2018 [14] and has been completely re-implemented. The implementation is now based on the *Matching EvaLuation Toolkit* [5,11] as well as the *KGvec2go* [12] REST API. A contribution of this paper is also an extension to the MELT framework in the form of a KGvec2go Java client available in the MELT-ML module [6] of MELT 2.6.

1.2 Specific Techniques Used

After the basic concepts of this matcher are introduced (*Foundations*), the specific techniques applied are presented.

Foundations

WebIsALOD Dataset A frequent problem that occurs when working with external background knowledge is the fact that less common entities are not contained within a knowledge base. The *WebIsA* [17] database is an attempt to tackle this problem by providing a dataset which is not based on a single source of knowledge – like *DBpedia* [8] – but instead on the whole Web: The dataset consists of hypernymy relations extracted from the *Common Crawl*[4], a freely downloadable crawl of a significant portion of the Web. A sample triple from the dataset is *european_union skos:broader international_organization*[5]. The dataset is also available via a Linked Open Data (LOD) endpoint[6] under the name *WebIsALOD* [4]. In the LOD dataset, a machine-learned confidence score $c \in [0, 1]$ is assigned to every hypernymy triple indicating the assumed degree of truth of the statement.

RDF2Vec The background dataset can be viewed as a very large knowledge graph; in order to obtain a similarity score for nodes and edges in that graph, the *RDF2Vec* [16] approach is used. It applies the *word2vec* [9,10] model to RDF data: Random walks are performed for each node and are interpreted as sentences. After the walk generation, the sentences are used as input for the word2vec algorithm. As a result, one obtains a vector for each word, i.e., a concept in the RDF graph. Multiple flavors of RDF2Vec have been developed in the past such as biased walks [1] or *RDF2Vec Light* [13].[7]

KGvec2go Training embeddings on large knowledge graphs can be computationally very expensive. Moreover, the resulting embedding models can be very large since a multidimensional vector needs to be persisted for every node in the knowledge graph. However, most downstream applications require only a small subset of node vectors. The *KGvec2go* project [12] addresses these problems by providing a free REST API[8] for pre-trained RDF2Vec models on various large knowledge graphs (among which WebIsALOD is also available).

Monolingual Matching *ALOD2Vec Matcher* is a monolingual matching system. For the alignment process, the system retrieves the labels of all elements of the ontologies to be matched. A filter adds all simple string matches to the final alignment in order to increase the performance. The remaining labels are linked to concepts in the background dataset, are compared, and the best solution is added to the final alignment. A high-level view of the matching system is provided in Figure 1.

[4] see `http://commoncrawl.org/`
[5] see `http://webisa.webdatacommons.org/concept/european_union_`
[6] see `http://webisa.webdatacommons.org/`
[7] For a good overview of the RDF2Vec approach and its applications, refer to `http://www.rdf2vec.org/`
[8] see `http://kgvec2go.org/api.html`

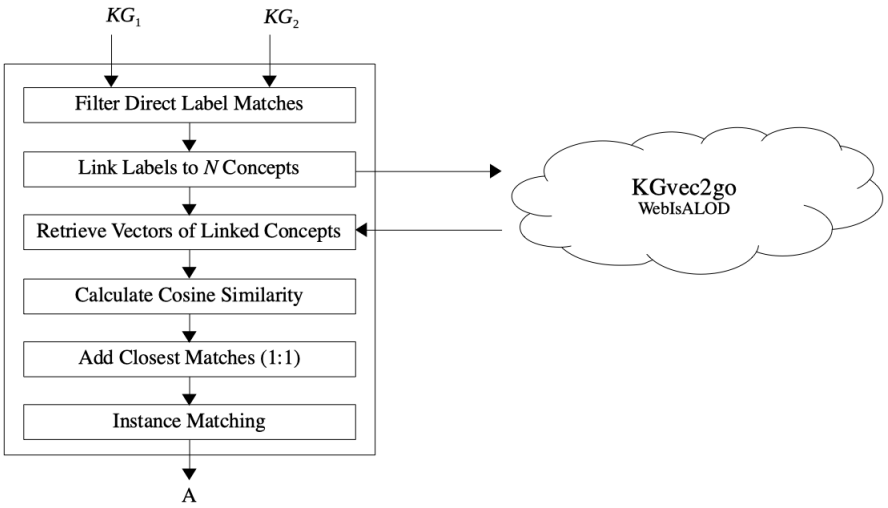


Fig. 1. High-level view of the ALOD2Vec matching process. KG_1 and KG_2 represent the input ontologies and optionally instances. The final alignment is referred to as A.

The first step is to link the obtained labels from the ontology to concepts in the WebIsALOD dataset. Therefore, string operations are performed on the label and it is checked whether the label is available in WebIsALOD. If it cannot be found, a token-lookup is performed. Given two entities e_1 and e_2, the matcher uses their textual labels to link them to concepts e'_1 and e'_2 in the external dataset. Afterwards, the embedding vectors $v_{e'_1}$ and $v_{e'_2}$ of the linked concepts (e'_1 and e'_2) are retrieved via a Web request and the cosine similarity between those is calculated. Hence: $sim(e_1, e_2) = sim_{cosine}(v_{e'_1}, v_{e'_2})$. If $sim(e_1, e_2) > t$ where t is a threshold in the range of 0 and 1, a correspondence is added to a temporary alignment. In a last step, a one-to-one arity is enforced by applying a *Maximum Weight Bipartite* [2] filter on the temporary alignment.

In order to consume the vectors in Java, a client has been implemented and contributed to the MELT-ML module. The KGvec2go REST API can now be accessed though class `KGvec2goClient`. Even though this matcher only uses the WebIsALOD dataset, the implementation supports all datasets accessible on KGvec2go. The extension is available by default in MELT 2.6.

Instance Matching For the 2020 version of the matching system, an instance matching module has been added. After classes and properties have been matched, instances are matched using a string index. The confidence score assigned to instances belonging to matched classes is higher than that of matches between instances belonging to non-matched classes.

Explainability *ALOD2Vec Matcher* provices an explanation for every correspondence that is added to the final alignment. Therefore, the extension capa-

bilities of the alignment format [3] are used. Two concrete examples from the *Anatomy track* for explanations of the matching system are: "Label 'aqueous humour' of ontology 1 and label 'Aqueous Humor' of ontology 2 have a very similar writing." or "The following two label sets have a cosine above the given threshold: |lens|anterior|epithelium| and |anterior|surface|lens|". In order to explain a correspondence, the `description` property[9] of the *Dublin Core Metadata Initiative* is used.

1.3 Extensions to the Matching System for the 2020 Campaign

The 2020 system has been completely rewritten. Among the significant changes are an improved handling of string matches, an instance matching module for the *knowledge graph track* [7], explanations on the level of correspondences, a simplified linking process as well as the usage of a Web endpoint compared to a local key value database that has been used before. It is important to note that the 2020 system uses the KGvec2go model for ALOD2Vec which is not equal to the model trained in 2018. Due to the usage of the KGvec2go API, the SEALS package is now several magnitudes smaller than before in terms of required disk space.[10] The smaller package cost comes at the price of a slower system runtime due to API calls. However, this matcher still scored at the exact median of all matching systems in terms of runtime on the anatomy track this year. The 2020 implementation is publicly available on GitHub.[11]

2 Results

2.1 Anatomy Track

On the anatomy dataset, the recall could be significantly improved in 2020 compared to the 2018 version of the matching system. Despite a drop in precision, the new *ALOD2Vec Matcher* achieves an overall higher F_1 score. Due to multiple API calls to KGvec2go, the runtime performance decreased compared to the 2018 version of the matcher.

2.2 Conference Track

On the conference track, the new matcher configuration achieved a better result than the 2018 one in terms of F_1 due to a higher recall (from 0.5 in 2018 to 0.52 in 2020). The overall F_1 score on `ra1-M3` was 0.59.

[9] see `http://purl.org/dc/terms/description`

[10] The 2018 version of the matching system had to be submitted via a download link due to its large size. The 2020 version was submitted using the default process.

[11] see `https://github.com/janothan/ALOD2VecMatcher`

2.3 Knowledge Graph Track

This is the first year that *ALOD2Vec Matcher* participates in the knowledge graph track. The system could complete all matching tasks in time. Due to the new instance matching module, this matcher obtains the second best results achieving almost the same score as the *Wiktionary Matcher 2020* [15]. The overall F_1 score was 0.87 on the complete track.

3 Conclusion

In this paper, we presented the newest version of the *ALOD2Vec Matcher*, a matcher utilizing an RDF2Vec vector representation of the WebIsALOD dataset, as well as its results in the 2020 OAEI. The matching system has been improved compared to its 2018 version. *ALOD2Vec Matcher* now uses a remote vector API which makes the matcher package very portable due to its substantially reduced size. Overall, the results of the matching system could be significantly improved compared to its last OAEI participation and is the second best performing system on the knowledge graph track.

References

1. Cochez, M., Ristoski, P., Ponzetto, S.P., Paulheim, H.: Biased graph walks for RDF graph embeddings. In: Akerkar, R., Cuzzocrea, A., Cao, J., Hacid, M. (eds.) Proceedings of the 7th International Conference on Web Intelligence, Mining and Semantics, WIMS 2017, Amantea, Italy, June 19-22, 2017. pp. 21:1–21:12. ACM (2017). https://doi.org/10.1145/3102254.3102279, https://doi.org/10.1145/3102254.3102279
2. Cruz, I.F., Antonelli, F.P., Stroe, C.: Efficient selection of mappings and automatic quality-driven combination of matching methods. In: Proceedings of the 4th International Conference on Ontology Matching-Volume 551. pp. 49–60. Citeseer (2009)
3. David, J., Euzenat, J., Scharffe, F., dos Santos, C.T.: The alignment API 4.0. Semantic Web **2**(1), 3–10 (2011). https://doi.org/10.3233/SW-2011-0028, https://doi.org/10.3233/SW-2011-0028
4. Hertling, S., Paulheim, H.: Webisalod: Providing hypernymy relations extracted from the web as linked open data. In: d'Amato, C., Fernández, M., Tamma, V.A.M., Lécué, F., Cudré-Mauroux, P., Sequeda, J.F., Lange, C., Heflin, J. (eds.) The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part II. Lecture Notes in Computer Science, vol. 10588, pp. 111–119. Springer (2017). https://doi.org/10.1007/978-3-319-68204-4_11, https://doi.org/10.1007/978-3-319-68204-4_11
5. Hertling, S., Portisch, J., Paulheim, H.: MELT - matching evaluation toolkit. In: Acosta, M., Cudré-Mauroux, P., Maleshkova, M., Pellegrini, T., Sack, H., Sure-Vetter, Y. (eds.) Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9-12, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11702, pp. 231–245. Springer (2019). https://doi.org/10.1007/978-3-030-33220-4_17, https://doi.org/10.1007/978-3-030-33220-4_17

6. Hertling, S., Portisch, J., Paulheim, H.: Supervised ontology and instance matching with MELT. In: OM@ISWC 2020 (2020), to appear
7. Hofmann, A., Perchani, S., Portisch, J., Hertling, S., Paulheim, H.: Dbkwik: Towards knowledge graph creation from thousands of wikis. In: Nikitina, N., Song, D., Fokoue, A., Haase, P. (eds.) Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017. CEUR Workshop Proceedings, vol. 1963. CEUR-WS.org (2017), http://ceur-ws.org/Vol-1963/paper540.pdf
8. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., Bizer, C.: Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. Semantic Web **6**(2), 167–195 (2015). https://doi.org/10.3233/SW-140134, https://doi.org/10.3233/SW-140134
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Bengio, Y., LeCun, Y. (eds.) 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings (2013), http://arxiv.org/abs/1301.3781
10. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Burges, C.J.C., Bottou, L., Ghahramani, Z., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States. pp. 3111–3119 (2013), http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality
11. Portisch, J., Hertling, S., Paulheim, H.: Visual analysis of ontology matching results with the MELT dashboard. In: The Semantic Web: ESWC 2020 Satellite Events (2020)
12. Portisch, J., Hladik, M., Paulheim, H.: Kgvec2go - knowledge graph embeddings as a service. In: Calzolari, N., Béchet, F., Blache, P., Choukri, K., Cieri, C., Declerck, T., Goggi, S., Isahara, H., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S. (eds.) Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020. pp. 5641–5647. European Language Resources Association (2020), https://www.aclweb.org/anthology/2020.lrec-1.692/
13. Portisch, J., Hladik, M., Paulheim, H.: Rdf2vec light - a lightweight approach for knowledge graph embeddings. In: Proceedings of the ISWC 2020 Posters Demonstrations (2020), to appear
14. Portisch, J., Paulheim, H.: Alod2vec matcher. In: OM@ISWC. CEUR Workshop Proceedings, vol. 2288, pp. 132–137. CEUR-WS.org (2018)
15. Portisch, J., Paulheim, H.: Wiktionary Matcher results for OAEI 2020. In: OM@ISWC 2020 (2020), to appear
16. Ristoski, P., Rosati, J., Noia, T.D., Leone, R.D., Paulheim, H.: Rdf2vec: RDF graph embeddings and their applications. Semantic Web **10**(4), 721–752 (2019). https://doi.org/10.3233/SW-180317, https://doi.org/10.3233/SW-180317
17. Seitner, J., Bizer, C., Eckert, K., Faralli, S., Meusel, R., Paulheim, H., Ponzetto, S.P.: A large database of hypernymy relations extracted from the web. In: Calzolari, N., Choukri, K., Declerck, T., Goggi, S., Grobelnik, M., Maegaard, B., Mariani, J., Mazo, H., Moreno, A., Odijk, J., Piperidis, S. (eds.) Proceedings of

the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016. European Language Resources Association (ELRA) (2016), http://www.lrec-conf.org/proceedings/lrec2016/summaries/204.html

OAEI 2020 results for AML and AMLC

Beatriz Lima[1], Daniel Faria[2], Francisco M. Couto[1],
Isabel F. Cruz[3], and Catia Pesquita[1]

[1] LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal
[2] BioData.pt & INESC-ID, Lisboa, Portugal
[3] ADVIS Lab, Department of Computer Science, University of Illinois at Chicago, USA

Abstract. AgreementMakerLight (AML) is a scalable and extensible ontology matching system with an alignment repair functionality and a strong focus on the use of external knowledge. In OAEI 2020, AML's development focused mainly on expanding its range of complex matching algorithms, but there were also improvements on its instance matching pipeline and on its ontology parsing algorithm. AML remains the system with the broadest coverage of OAEI tracks, and among the top performing systems overall.

1 Presentation of the System

1.1 State, Purpose, General Statement

AgreementMakerLight (AML) is an ontology matching system inspired by AgreementMaker [1, 2, 10] but designed anew to tackle the matching of very large ontologies efficiently [7]. It is a general purpose system that is able to successfully tackle problems across the whole spectrum of ontology matching, irrespective of their domain.
AML is primarily based on lexical matching algorithms [8], but also includes structural algorithms for both matching and filtering, as well as its own logical repair algorithm [9]. It is capable of using external background knowledge, and even automatically selecting background knowledge sources for any given ontologies to match [6].
AMLC is a new version of AML developed to tackle complex ontology matching. At this time, it remains separate from the main AML codebase and OAEI submission, but we aim to merge the two versions in the near future.
This year, our development focused mainly on the implementation of pattern mining ontology matching algorithms in AMLC, based on association rules and inspired by the work of Zhou et al. [11]. As of our OAEI submission, AMLC included only variants of these algorithms for detecting simple class and property mappings, but we are in the process of implementing variants for complex mappings.
As has been the case in recent years, we also participated in the SPIMBENCH and Link Discovery tracks via the HOBBIT platform. In the case of SPIMBENCH, we participated with the HOBBIT adaptation of the main AML codebase. In the case of Link Discovery, we participated with a specialized version of AML, AML-Spatial, due to

the unique characteristics of the matching tasks in this track and to the unavailability of the TBox assertions in the HOBBIT datasets.

1.2 Specific Techniques Used

This section describes only the features of AML that are new for OAEI 2020. It also describes AMLC, a variant of AML tailored to complex matching. For further information on AML's simple matching strategy, please consult AML's original paper [7] as well as the AML OAEI results publications of 2016-2018 [4, 3, 5].

Our main development this year was a modular association rule mining framework for ontology matching, inspired by the work of Zhou et al. [11]. This strategy resembles the common market basket analysis, where we take into account how frequently two entities of different ontologies are related to common instances, given a populated dataset. Our framework features a central association rule mining algorithm implementation that selects patterns (i.e., mappings) based on their confidence and support, and a suite of algorithms devoted to finding individual types of patterns and computing their confidence and support from among the set of instances. As of the OAEI submission we had implemented only algorithms for detecting simple class and property mappings, but we are in the process of implementing algorithms for each type of complex mapping.

1.3 Adaptations Made for the Evaluation

As has been the case in recent OAEI editions, the Link Discovery submission of AML is adapted to these particular tasks and datasets, as their specificities (namely the absence of a Tbox) demand a dedicated submission. The same is also true to some extent of AML's Complex Matching submission.
As usual, our submission included precomputed dictionaries with translations, to circumvent Microsoft® Translator's query limit.

1.4 Link to the System and Parameters File

AML is an open source ontology matching system and is available through GitHub: https://github.com/AgreementMakerLight.

2 Results

AML's OAEI 2020 results are summarized in Table 1 and discussed in the following subsections.

2.1 Anatomy

AML had a 0.7% increase in precision and a 0.9% decrease in recall, resulting in a 0.2% decrease in F-measure, in comparison with its performance in recent years. These differences are an unexpected consequence of minor changes in AML's general configuration.

Table 1: Summary of OAEI 2020 results for AML and AMLC.

Task	Precision	Recall/ Coverage	F-measure	Run time (s)	Rank[1]
—— Anatomy ——					
Mouse-Human	0.956	0.927	0.941	29	1
—— Biodiversity & Ecology ——					
FLOPO-PTO	0.766	0.820	0.792	53.7	3
ENVO-SWEET	0.810	0.927	0.865	38.8	1
ANAEETHES-GEMET	0.976	0.764	0.857	4.2	3
AGROVOC-NALT	0.955	0.835	0.890	139.5	1[a]
—— Complex ——					
Conference	0.31	0.37	0.34	-	1[a]
Populated Conference	0.23-0.51	0.26-0.31	N/A	-	N/A
Hydrography	0.45	0.05	0.10	-	1[b]
Geolink	0.50	0.23	0.32	-	2
Populated Geolink	0.50	0.23	0.32	-	4
Populated Enslaved	0.73	0.28	0.40	-	1
Taxon	0.19-0.40	0	N/A	-	N/A
—— Conference ——					
OntoFarm (ra1-M3)	0.84	0.66	0.74	-	1
OntoFarm (ra2-M3)	0.82	0.61	0.70	-	1
OntoFarm (rar2-M3)	0.78	0.62	0.69	-	2
OntoFarm (Discrete)	0.79	0.77	0.78	-	1
OntoFarm (Continuous)	0.80	0.74	0.77	-	1
DBpedia-OntoFarm	0.48	0.67	0.56	-	1
—— Disease & Phenotype ——					
HP-MP	0.910	0.79	0.816	102	3
DOID-ORDO	0.682	0.834	0.750	200	2
—— Interactive Matching ——					
Anatomy (error 0.0)	0.972	0.933	0.952	37.3	1
Anatomy (error 0.1)	0.962	0.929	0.945	37.5	1
Anatomy (error 0.2)	0.951	0.928	0.939	37.4	1
Anatomy (error 0.3)	0.942	0.924	0.933	37.2	1
Conference (error 0.0)	0.91	0.698	0.79	30.1	2
Conference (error 0.1)	0.843	0.682	0.754	30	1
Conference (error 0.2)	0.777	0.677	0.723	30.3	1
Conference (error 0.3)	0.721	0.65	0.684	30.5	1
—— Knowledge Graph ——					
Aggregate (class)	0.98	0.81	0.89	–	1
Aggregate (property)	0.92	0.57	0.70	–	6
Aggregate (instance)	0.90	0.80	0.85	–	3[b]
Aggregate (all)	0.90	0.80	0.85	3055	3[b]
—— Large Biomedical Ontologies ——					
FMA-NCI small	0.958	0.91	0.933	38	1

FMA-NCI whole	0.806	0.881	0.842	82	1
FMA-SNOMED small	0.923	0.762	0.835	101	1
FMA-SNOMED whole	0.685	0.710	0.697	181	3
SNOMED-NCI small	0.906	0.746	0.818	629	1
SNOMED-NCI whole	0.862	0.687	0.765	381	1
	—— Link Discovery ——				
Spatial (mainbox)	1.0	1.0	1.0	11172	1^b
	—— Multifarm ——				
Different Ontologies	0.72	0.35	0.47	170	1
Same Ontologies	0.94	0.28	0.17	–	2
	—— SPIMBENCH ——				
SPIMBENCH (mainbox)	0.839	0.884	0.860	38772	4

[1]according to F-measure; [a] only system with results; [b] tied with other systems

2.2 Biodiversity and Ecology

AML improved its results on both the FLOPO-PTO and the ENVO-SWEET tasks in comparison with last year. It was surpassed by two versions of LogMap on the FLOPO-PTO task, but remained the best performing system in the ENVO-SWEET task.
With respect to the new tasks, AML ranked third in the ANAEETHES-GEMET task, and was the only system able to produce results in the AGROVOC-NALT task.

2.3 Complex Matching

AMLC was one of three tools able to generate complex correspondences, and the only tool able to produce results in the (non-populated) Conference task, which uses the simple reference alignment as input. While its performance was among the best in most tasks, it remains mediocre in comparison with its performance in simple matching tasks, underpinning the fact that there is much room for improvement in complex ontology matching.
We unfortunately were unable to finish implementing the suite of pattern mining algorithms for complex ontology matching in time for this OAEI edition, which likely would have improved AML's performance substantially in populated complex tasks.

2.4 Conference

AML had the exact same results as in recent years, with F1-measures of 74% according to the full reference alignment (ra1), 70% according to the extended reference alignment (ra2), 78% according to the discrete uncertain reference alignment, and 77% according to the continuous one, ranking first in all four evaluation variants. It ranked second in the evaluation with the violation free version of the extended reference alignment (rar2), likely because AML's repair algorithm deliberately does not address conservativity violations, as we do not subscribe to conservativity as a guiding principle in ontology matching.

AML was one of only five systems able to participate in a new unannounced task consisting in matching the DBpedia to the OntoFarm ontologies, and had the highest F-measure among those five.

2.5 Disease and Phenotype

AML ranked it third and second in F-measure in the HP-MP and DOID-ORDO tasks, respectively. However, as has been the trend, AML was one of the systems with the highest number of unique mappings (i.e., mappings not proposed by any other system). Since the evaluation in this track is based on a 3-vote consensus alignment, rather than a true reference alignment, and unique mappings are not otherwise assessed, this severely affects AML's evaluation, making its results below average in comparison with other biomedical matching tasks.

2.6 Interactive Matching

AML had a lower performance than last year in the Anatomy track, undoubtedly tied to its change in performance in the non-interactive version of the track. Its results in the Conference track remained the same. Overall it remains the interactive system that is the least impacted by the oracle errors.

2.7 Large Biomedical Ontologies

AML's performance in this track was similar to last year's, but with decimal increases in F-measure across all tasks, likely due to the same changes that affected its performance in the Anatomy track. It remains the best performing system in five out of the six tasks.

2.8 Knowledge Graph

Contrarily to last year, AML was able to complete all of the five tasks in a timely manner, having a global F-measure of 0.85, which ranked it third overall. It had the best performance in matching classes.

2.9 Link Discovery

As in previous years, AML and all other participants produced a perfect result (100% F-measure) in the Spatial track. AML had the highest run time among participating systems, though this was not true in all tasks.

2.10 Multifarm

AML's results were slightly better than last years', with a 2% increase in F-measure in the different ontologies modality and a 1% increase in the same ontologies modality. These differences are due to correcting a minor configuration problem when using AML's word-matching algorithm in a multilingual setting.

2.11 SPIMBENCH

AML obtained the same results as last year, with an F-measure of 86%, which ranked it fourth.

3 General Comments on the Results

In 2020, AML was once again the system that tackled the most OAEI tracks and datasets, and maintained its status as one of best performing and broadest matching systems competing in the OAEI.
Nonetheless, there is still some work to be done in terms of complex matching, in order to be able to provide more robust results. We will strive to refine and improve AML's complex matching pipeline, particularly by upgrading our association rule based approach.

4 Conclusions

Like in recent years, AML was the matching system that participated in the most OAEI tracks and datasets, and it was among the top performing systems in most of them. AML's performance was very similar to those of recent years in any of the long-standing OAEI tracks, as most of our development effort went into tackling new challenges, such as pattern mining approaches for complex matching.
Complex matching remains one of the biggest challenges in ontology matching, and will remain the main focus of AML's development in the near future.

Acknowledgments

DF was funded by the Portuguese FCT Grant 22231 BioData.pt (co-financed by FEDER). CP and BL are supported by FCT through project SMILAX (PTDC/EEI-ESS/4633/2014), and the LASIGE Research Unit (UIDB/00408/2020 and UIDP/00408/2020).

FMC was also funded by PTDC/CCI-BIO/28685/2017. The research of IFC was partially funded by NSF award III-1618126 and by NIGMS-NIH award R01GM125943.

References

1. I. F. Cruz, F. Palandri Antonelli, and C. Stroe. AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. *PVLDB*, 2(2):1586–1589, 2009.
2. I. F. Cruz, C. Stroe, F. Caimi, A. Fabiani, C. Pesquita, F. M. Couto, and M. Palmonari. Using AgreementMaker to Align Ontologies for OAEI 2011. In *ISWC International Workshop on Ontology Matching (OM)*, volume 814 of *CEUR Workshop Proceedings*, pages 114–121. CEUR-WS.org, 2011.

3. D. Faria, B. S. Balasubramani, V. R. Shivaprabhu, I. Mott, C. Pesquita, F. M. Couto, and I. F. Cruz. Results of AML in OAEI 2017. In *ISWC International Workshop on Ontology Matching (OM)*, volume 2032 of *CEUR Workshop Proceedings*, pages 122–128. CEUR-WS.org, 2017.
4. D. Faria, C. Pesquita, B. S. Balasubramani, C. Martins, J. Cardoso, H. Curado, F. M. Couto, and I. F. Cruz. OAEI 2016 results of AML. In *ISWC International Workshop on Ontology Matching (OM)*, volume 1766, pages 138–145. CEUR-WS.org, 2016.
5. D. Faria, C. Pesquita, B. S. Balasubramani, T. Tervo, D. Carriço, R. Garrilha, F. M. Couto, and I. F. Cruz. Results of AML Participation in OAEI 2018. In *ISWC International Workshop on Ontology Matching (OM)*, volume 2288 of *CEUR Workshop Proceedings*, pages 125–131. CEUR-WS.org, 2018.
6. D. Faria, C. Pesquita, E. Santos, I. F. Cruz, and F. M. Couto. Automatic Background Knowledge Selection for Matching Biomedical Ontologies. *PLoS One*, 9(11):e111226, 2014.
7. D. Faria, C. Pesquita, E. Santos, M. Palmonari, I. F. Cruz, and F. M. Couto. The AgreementMakerLight Ontology Matching System. In *OTM Conferences - ODBASE*, pages 527–541, 2013.
8. C. Pesquita, D. Faria, C. Stroe, E. Santos, I. F. Cruz, and F. M. Couto. What's in a "nym"? Synonyms in Biomedical Ontology Matching. In *International Semantic Web Conference (ISWC)*, pages 526–541, 2013.
9. E. Santos, D. Faria, C. Pesquita, and F. M. Couto. Ontology Alignment Repair Through Modularization and Confidence-based Heuristics. *PLoS ONE*, 10(12):e0144807, 2015.
10. W. Sunna and I. F. Cruz. In *International Conference on GeoSpatial Semantics (GeoS)*, pages 82–97. Springer.
11. L. Zhou, M. Cheatham, and P. Hitzler. Towards Association Rule-Based Complex Ontology Alignment. In X. Wang, F. A. Lisi, G. Xiao, and E. Botoeva, editors, *Semantic Technology*, pages 287–303, Cham, 2020. Springer International Publishing.

AROA Results for OAEI 2020*

Lu Zhou and Pascal Hitzler

DaSe Lab, Kansas State University, Manhattan KS 66506, USA
{luzhou, hitzler}@ksu.edu

Abstract. This paper introduces the results of an ontology alignment system named Association Rule-based Ontology Alignment (AROA) in the Ontology Alignment Evaluation Initiative (OAEI) 2020 campaign. This ontology alignment system focuses on producing simple and complex alignment between ontologies that are populated with instance data. This is the second participation of AROA in the OAEI campaign, and it produces the best performance in terms of relaxed F-measure on two benchmarks in complex track, which are populated GeoLink and populated Enslaved.

1 Presentation of the system

1.1 State, purpose, general statement

AROA (Association Rule-based Ontology Alignment) system aims to automatically generate simple and complex alignment between two and more ontologies. These ontologies are required to have shared common instance data because AROA relies on association rule mining and requires these instances as input to discover interesting relations. After generating a set of association rules, AROA utilizes the simple and complex correspondence patterns that have been widely accepted in the Ontology Matching community [4, 5] to further narrow a large number of rules down to more meaningful ones and finally establishes the alignments.

1.2 Specific techniques used

Figure 1 illustrates the overview of AROA alignment system. In this section, we introduce each step of AROA alignment system along with some concepts that we frequently use in the AROA system, such as association rule mining, FP-growth algorithm, and complex alignment generation.

Clean Triple. First, AROA extracts all triples as the format of ⟨Subject, Predicate, Object⟩ from the source and target ontologies. Each item in a triple is expressed as a web URI. After collecting all of the triples, we clean the data based on the following criteria: we only keep the triples that contain at least one entity under the source or the target ontology namespace or the triples contain rdf:type information, as our algorithm relies on this information.

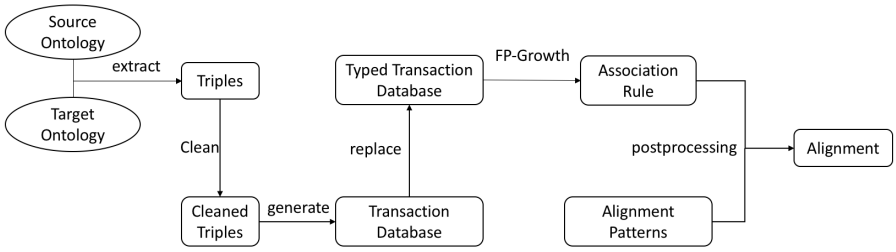


Fig. 1. Overview of AROA Alignment System

Generate Transaction Database. After the filtering process, we generate the transaction database as the input for the FP-growth algorithm. Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of distinct attributes called items. Let $D = \{t_1, t_2, \ldots, t_m\}$ be a set of transactions where each transaction in D has a unique transaction ID and contains a subset of the items in I. Table 1 shows a list of transactions corresponding to a list of triples. Instance data can be displayed as a set of triples, each consisting of subject, predicate, and object. Here, subjects represent the identifiers and the set of corresponding properties with the objects represent transactions, which are separated by the symbol "|". I.e., a transaction is a set $T = (s, Z)$ such that s is a subject, and each member of Z is a pair (p, o) of a property and an object such that (s, p, o) is an instance triple.

Generate Typed Transaction Database. Then we replace the object in the triples with its rdf:type[1] because we focus on generating schema-level (rather than instance-level) mapping rules between two ontologies, and the type

[1]If there are multiple types of the object, it can also combine the subject and predicate as additional information to determine the correct type, or keep both types as two triples.

Table 1. Triples and Corresponding Transactions

$s_1\ p_1\ o_1$
$s_1\ p_2\ o_2$
$s_1\ p_4\ o_4$
$s_2\ p_1\ o_1$
$s_2\ p_2\ o_2$
$s_2\ p_3\ o_3$
$s_2\ p_4\ o_4$
$s_3\ p_1\ o_1$
$s_3\ p_2\ o_2$

TID	Itemsets
s_1	$p_1\|o_1,\ p_2\|o_2,\ p_4\|o_4$
s_2	$p_1\|o_1,\ p_2\|o_2,\ p_3\|o_3,\ p_4\|o_4$
s_3	$p_1\|o_1,\ p_2\|o_2$

Table 2. Original Transaction Database

TID	Itemsets
x_1	gbo:hasAward|y_1, gmo:fundedBy|y_2
x_2	gbo:hasFullName|y_3, gmo:hasPersonName|y_4
x_3	rdf:type|gbo:Cruise, rdf:type|gmo:Cruise

Table 3. Typed Transaction Database

TID	Itemsets
x_1	gbo:hasAward|gbo:Award, gmo:fundedBy|gmo:FundingAward
x_2	gbo:hasFullName|xsd:string, gmo:hasPersonName|gmo:PersonName
x_3	rdf:type|gbo:Cruise, rdf:type|gmo:Cruise

information of the object is more meaningful than the original URI. If an object in a triple has rdf:type of a class in ontology, we replace the URI of the object with its class. If the object is a data value, the URI of the object is replaced with the datatype. If the object already is a class in ontology, it remains unchanged. Tables 2 and 3 show some examples of the conversion.

Generate Association Rules. Our alignment system mainly depends on a data mining algorithm called association rule mining, which is a rule-based machine learning method for discovering interesting relations between variables in large databases [3]. Many algorithms for generating association rules have been proposed, like Apriori [1] and FP-growth algorithm [2]. In this paper, we use FP-growth to generate association rules between ontologies, since the FP-growth algorithm has been proven superior to other algorithms [2]. The FP-growth algorithm is run on the transaction database in order to determine which combinations of items co-occur frequently. The algorithm first counts the number of occurrences of all individual items in the database. Next, it builds an FP-tree structure by inserting these instances. Items in each instance are sorted by descending order of their frequency in the dataset so that the tree can be processed quickly. Items in each instance that do not meet the predefined thresholds, such as minimum support and minimum confidence (see below for these terms), are discarded. Once all large itemsets have been found, the association rule creation begins. Every association rule is composed of two sides. The left-hand-side is called the antecedent, and the right-hand-side is the consequent. These rules indicate that whenever the antecedent is present, the consequent is likely to be

Table 4. Examples of Association Rules

Antecedent	Consequent
$p_4\|o_4$, $p_1\|o_1$	$p_2\|o_2$
$p_2\|o_2$	$p_1\|o_1$
$p_4\|o_4$	$p_1\|o_1$

Table 5. The Alignment Pattern Types Covered in AROA System

Pattern	Category
Class Equivalence	1:1
Class Subsumption	1:1
Property Equivalence	1:1
Property Subsumption	1:1
Class by Attribute Type	1:n
Class by Attribute Value	1:n
Property Typecasting Equivalence	1:n
Property Typecasting Subsumption	1:n
Typed Property Chain Equivalence	m:n
Typed Property Chain Subsumption	m:n

as well. Table 4 shows some examples of association rules generated from the transaction database in Table 1.

Generate Alignment. AROA utilizes some simple and complex correspondences that have been widely accepted in Ontology Matching community to further filter rules [4, 5] and finally generate the alignments. There are a total of 10 different types of correspondences that AROA covers this year. Table 5 lists all the simple and complex alignment correspondences and corresponding categories. Since the association rule mining might generate a large number of rules, in order to narrow the association rules down to a smaller set, AROA follows these patterns to generate corresponding alignments. For example, Class by Attribute Type (CAT) is a classic complex alignment pattern. This type of pattern was first introduced in [4]. It states that a class in the source ontology is in some relationship to a complex construction in the target ontology. This complex construction may comprise an object property and its range. Class C_1 is from ontology O_1, and object property op_1 and its range t_1 are from ontology O_2.

Association Rule format: $\mathsf{rdf{:}type}|C_1 \rightarrow \mathsf{op_1}|\mathsf{t_1}$
Example: $\mathsf{rdf{:}type}|\mathsf{gbo{:}PortCall} \rightarrow \mathsf{gmo{:}atPort}|\mathsf{gmo{:}Place}$
Generated Alignment: $\mathsf{gbo{:}PortCall(x)} \rightarrow \mathsf{gmo{:}atPort(x,y)} \wedge \mathsf{gmo{:}Place(y)}$

In this example, this association rule implies that if the subject x is an individual of class gbo:PortCall, then x is subsumed by the domain of gmo:atPort with its range gmo:Place. The equivalence relationship can be generated by combining another association rule holding the reverse information. Other simple and complex alignments are also generated by following the same steps.

1.3 Adaptations made for the evaluation

AROA is an instance-based ontology alignment system. Therefore, AROA embeds Apache Jena Fuseki server in the system. The ontologies are first downloaded from the SEALS repository. And then, AROA uploads and stores the

Table 6. The Number of Alignments Found on Populated GeoLink Benchmark

Alignment Patterns	Category	Reference Alignment	AROA	
-	-	-	# of Correct Entities	# of Correct Relation
Class Equiv.	1:1	10	10	10
Class Subsum.	1:1	2	1	0
Property Equiv.	1:1	7	5	5
Property Typecasting Subsum.	1:n	5	3	0
Property Chain Equiv.	m:n	26	15	13
Property Chain Subsum.	m:n	17	7	0

ontologies in the embedded Fuseki server, which might take some time for this step to load large-size ontology pairs.

2 Results

This year, AROA alignment system evaluates its performance on the populated GeoLink benchmark [5, 6] and populated Enslaved benchmark [7]. In the populated GeoLink benchmark, there are 19 simple mappings, including 10 class equivalence, 2 class subsumption, and 7 property equivalence. And there are 48 complex mappings, including 5 property subsumption, 26 property chain equivalence, and 17 property chain subsumption. In the populated Enslaved benchmark, 15 simple mappings are all class equivalences. And there are 83 complex mappings, including 68 property chain equivalence and 15 property chain subsumption. Table 6 and Table 7 list the alignment patterns and categories in the populated GeoLink and populated Enslaved Benchmark with the results of AROA system. We list the numbers of identified mappings for each pattern. There are two dimensions that we can look into the details to understand the performance. The first dimension is the entity identification, which means, given an entity in the source ontology, the system should be able to generate related entities in the target ontology. Another dimension is relationship identification, in which the system should detect the correct relationship between these entities, such as equivalence and subsumption. Therefore, we list the number of correct entities and the number of correct relationships in order to understand the strengths and weaknesses of the system. For example, In the Table 6, AROA correctly identifies all 1:1 class equivalence including entity and relationship. AROA also finds one class subsumption alignment, which is the class $PortCall$ in the GeoLink Base Ontology (GBO) is related to the class Fix in the GeoLink Modular Ontology (GMO). However, it outputs the relationship between

Table 7. The Number of Alignments Found on Populated Enslaved Benchmark

Alignment Patterns	Category	Reference Alignment	AROA	
-	-	-	# of Correct Entities	# of Correct Relation
Class Equiv.	1:1	15	11	11
Property Chain Equiv.	m:n	68	29	29
Property Chain Subsum.	m:n	15	3	0

Table 8. The Performance Comparison on Populated GeoLink and Populated Enslaved Benchmarks

Matcher	Populated GeoLink						Populated Enslaved					
	(1:1)	(1:n)	m:n	Relaxed_Precision	Relaxed_F-measure	Relaxed_Recall	(1:1)	(1:n)	m:n	Relaxed_Precision	Relaxed_F-measure	Relaxed_Recall
Reference Alignment	19	5	43	-	-	-	15	0	83	-	-	-
AMLC	13	0	0	0.50	0.32	0.23	**12**	0	18	0.73	0.40	0.28
AROA	**15**	3	**22**	0.87	**0.60**	**0.46**	11	0	**32**	**0.80**	**0.51**	**0.38**
CANARD	**15**	2	17	**0.89**	0.54	0.39	3	0	16	0.42	0.19	0.13

PortCall and *Fix* as equivalence, which it should be subsumption. Therefore, we count the number of correct entities as 1 and the number of correct relations as 0. This criterion is also applied to other patterns. In the Table 7, AROA detects 73% (11 out of 15) of the simple class equivalences and 38% (32 out of 83) of the complex mappings in the populated Enslaved benchmark. In addition, we compare the performance of AROA against other complex alignment systems in Table 8. AMLC, AROA, and CANARD are only three systems can produce complex relations on the complex benchmarks. AROA found the highest number of complex alignments and achieved the best performance in terms of relaxed recall and relaxed f-measure on both benchmarks.[2]

3 General comments

From the performance comparison, AMLC, AROA, and CANARD can generate almost correct complex alignment, which means some alignments found by these two systems may not be completely correct, but it can be easily improved by semi-automated fashion. For example, the system can produce correct entities that should be involved in a complex alignment, but it doesn't output the correct relationship. Another possible situation is that the system can detect the correct relationship but fails to find all the entities. Based on these situations, we will investigate the incorrect alignments and improve the algorithm to find the relationship and entities as accurately as possible.

4 Conclusions

This paper introduces the AROA ontology alignment system and its preliminary results in the OAEI 2020 campaign. This year, AROA evaluates its performance on populated GeoLink and populated Enslaved benchmarks and achieves the best performance in terms of relaxed recall and relaxed f-measure among the three complex alignment systems. We will continue to evaluate AROA on other benchmarks and improve the algorithm in the near future.

5 Acknowledgement

This work has been supported by the National Science Foundation under Grant No. 2033521, KnowWhereGraph: Enriching and Linking Cross-Domain Knowl-

[2] http://oaei.ontologymatching.org/2020/results/complex/geolink/index.html

edge Graphs using Spatially-Explicit AI Technologies and the Andrew W. Mellon Foundation through the Enslaved project (identifiers 1708-04732 and 1902-06575).

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Bocca, J.B., Jarke, M., Zaniolo, C. (eds.) VLDB'94, Proc. of 20th International Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile. pp. 487–499. Morgan Kaufmann (1994), http://www.vldb.org/conf/1994/P487.PDF
2. Han, J., et al.: Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Min. Knowl. Discov. **8**(1), 53–87 (2004). https://doi.org/10.1023/B:DAMI.0000005258.31418.83, https://doi.org/10.1023/B:DAMI.0000005258.31418.83
3. Piatetsky-Shapiro, G.: Discovery, analysis, and presentation of strong rules. In: Knowledge Discovery in Databases, pp. 229–248. AAAI/MIT Press (1991)
4. Ritze, D., Meilicke, C., Sváb-Zamazal, O., Stuckenschmidt, H.: A pattern-based ontology matching approach for detecting complex correspondences. In: Shvaiko, P., Euzenat, J., Giunchiglia, F., Stuckenschmidt, H., Noy, N.F., Rosenthal, A. (eds.) Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009. CEUR Workshop Proceedings, vol. 551. CEUR-WS.org (2009), http://ceur-ws.org/Vol-551/om2009_Tpaper3.pdf
5. Zhou, L., et al.: A complex alignment benchmark: Geolink dataset. In: Vrandecic, D., Bontcheva, K., Suárez-Figueroa, M.C., Presutti, V., Celino, I., Sabou, M., Kaffee, L., Simperl, E. (eds.) The Semantic Web - ISWC 2018 - 17th International Semantic Web Conference, Monterey, CA, USA, October 8-12, 2018, Proceedings, Part II. Lecture Notes in Computer Science, vol. 11137, pp. 273–288. Springer (2018). https://doi.org/10.1007/978-3-030-00668-6_17, https://doi.org/10.1007/978-3-030-00668-6_17
6. Zhou, L., Cheatham, M., Krisnadhi, A., Hitzler, P.: Geolink data set: A complex alignment benchmark from real-world ontology. Data Intell. **2**(3), 353–378 (2020). https://doi.org/10.1162/dint_a_00054, https://doi.org/10.1162/dint_a_00054
7. Zhou, L., Shimizu, C., Hitzler, P., Sheill, A.M., Estrecha, S.G., Foley, C., Tarr, D., Rehberger, D.: The enslaved dataset: A real-world complex ontology alignment benchmark using wikibase. In: d'Aquin, M., Dietze, S., Hauff, C., Curry, E., Cudré-Mauroux, P. (eds.) CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020. pp. 3197–3204. ACM (2020). https://doi.org/10.1145/3340531.3412768, https://doi.org/10.1145/3340531.3412768

ATBox Results for OAEI 2020

Sven Hertling[0000−0003−0333−5888] and Heiko Paulheim[0000−0003−4386−8195]

Data and Web Science Group, University of Mannheim, Germany
{sven,heiko}@informatik.uni-mannheim.de

Abstract. ATBox matcher is a scalable system for instance (Abox) and schema (Tbox) matching. It uses two pipelines for generating candidates for the schema and instance matching, and utilizes the schema matches to further improve the instance correspondences. Using a string blocking method, ATBox is able to align large ontologies and can run on OAEI tracks like largebio and knowledge graph. The results look promising, but further features for better finding correct instance matches can be developed.

Keywords: Ontology Matching · Knowledge Graph

1 Presentation of the system

Nearly all systems submitted to the Ontology alignment Evaluation Initiative (OAEI) are able to align ontologies, schemas, or Tboxes, as they are called in description logics (DL). On the other hand, there are more and more instance tracks like spimbench, link discovery, geolink cruise, and knowledge graph, matching instances, or Aboxes, becomes equally important. The matcher presented in this paper, called *ATBox*, focuses on both the Abox and Tbox.

Especially the knowledge graph track needs scalable systems which can deal with hundred of thousands of instances [4]. Thus, the basis of this matcher is a blocking approach, which focuses on high recall. Its result is succesively fine tuned to increase the precision. Given this design, ATBox is also able to match large knowledge graphs like DBpedia [1] or YAGO [6].

1.1 State, purpose, general statement

The overall matching strategy of ATBox is shown in figure 1. The Tbox and Abox have different processing pipelines but the correspondences are combined in the end to get the final alignment.

Tbox matching is applied for all classes and properties (`owl:ObjectProperty`, `owl:DatatypeProperty`, and `rdf:Property`). They are retrieved by the jena[1] methods *OntModel.listClasses()* and *OntModel.listAllOntProperties()*.

[1] https://jena.apache.org

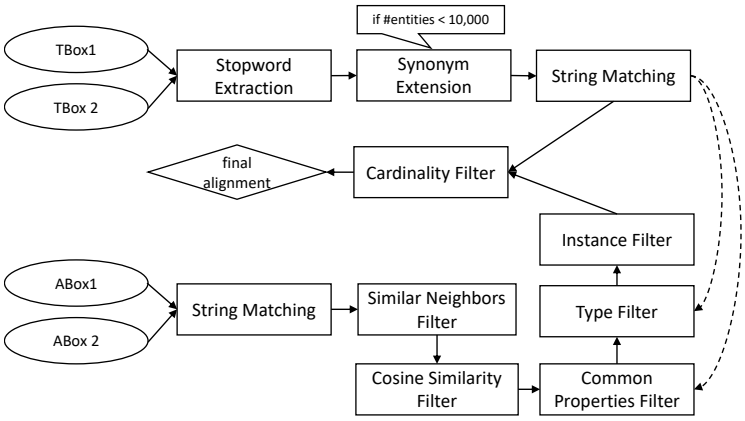


Fig. 1. Overview of the ATBox matcher strategy.

The Tbox matching (classes and properties) starts with the stopword extraction. In some cases the labels and/or fragments (which we define as the part after the last hashtag symbol # or slash /) contains tokens which appears very often like `class`, `infobox` etc. If such tokens appears in more than 20 % of all classes/properties (considered separately), then it is extracted as a corpus specific stop word. In case there are many such stop words, they are restricted to the five most occurring ones.

The synonyms (used during string matching) are extracted from the English Wiktionary to cover many different domains. The extraction is done with DBnary [8], a dataset containing Wiktionary as RDF. The extraction process starts with all resources of type dbnary:Page [2] within the English domain [3]. Then we follow the `describes` relation and extract all resources connected with property `synonym`. Furthermore we follow the relation `sense` to also find all the given senses and their synonyms. The lemmas are extracted directly from the URI.

Table 1. String processing steps in ATBox matcher for schema matches.

Processsing	Confidence	Levenshtein
equality	1.0	no
normalize	0.9	no
normalizeParentheses	0.8	no
defaultStopwords	0.7	no
corpusStopwords	0.6	yes
synonyms	0.5	no

[2] http://kaiko.getalp.org/dbnary#Page
[3] http://kaiko.getalp.org/dbnary/eng/

The string matching contains multiple different steps which are shown in table 1. All processing applies to `rdfs:label` and in case it is missing to the URI fragment. If the extracted text is exactly the same, the generated correspondence has a confidence of 1.0. During the normalization process, a word written in camel case[4] is separated with whitespace (e.g. `hasAge` to `has Age`) and afterwards lowercased. In case some UTF-8 characters are not normalized, we apply a normalization step for them (e.g. an accented character can be encoded in multiple different ways in UTF-8). All possible punctuations are furthermore removed and multiple whitespaces are combined into one. In case the normalized text matches, a confidence of 0.9 is assigned. In the `normalizeParentheses` step, all text within parentheses is removed. If the remaining normalized text (same as in `normalize` step) is equal, it assigns a confidence of 0.8. The reason behind is that many articles in KGs define concepts with same names to have the discriminating term in parentheses e.g. "Harry Potter (character)" and "Harry Potter (film series)". `DefaultStopwords` removes a given set of stopwords while keeping all other processing steps as before (confidence is 0.7). In the last processing step, the corpus specific stopwords, extracted before, are also removed and additionally allow a levenshtein distance[7] of 1 (but only in case the text is longer than 6 characters). In case it matches a correspondence with confidence of 0.6 is generated. If the amount of concepts are less than 10,000 for source and target, then a synonym step is added with a confidence of 0.5. In this step, the extracted synonyms are used to replace (possibly multiple) tokens with all available synonyms.

All string processing steps are executed in order starting with the highest confidence. If a match is found the remaining steps are also executed to find possible other candidates. As an example, a correspondence like <Harry_Potter,harry_potter, =, 0.9> is already found, then the processing continues and also add <Harry_Potter,Harry_Potter(Book), =, 0.8> to the resulting alignment.

The instance matching (Abox - shown in the lower part of the figure 1) starts directly with the string matching component. It reuses the processing steps described in the previous section without the corpus dependent stopword removal and synonym replacement. The applied steps are shown in table 2. The first four steps applies to the `rdfs:label` and if it is missing to the fragment of the URI. The confidence is decreasing with a step size of 0.1 starting with 1.0. In the second part, the additional properties `skos:prefLabel` and `skos:altLabel` are taken into account. If they match, the confidence is set to maximally 0.6 depending in which preprocessing step the match occurs. Once again, we allow matches which a lower confidence, even when a correspondence with a higher confidence is found. This increases the recall because it might be the case that the matched entity with a high confidence is not the best available match.

The string processing step generated an alignment with a high recall. All following steps try to increase the precision by generating additional confidences for each correspondence. This helps at the end of the processing pipeline to enforce a one to one alignment and selecting the right correspondence in

[4] https://en.wikipedia.org/wiki/Camel_case

Table 2. Processing steps for generating instance matches.

Processsing	Confidence	Property
equality	1.0	rdfs:label (or fragment)
normalize	0.9	rdfs:label (or fragment)
normalizeParentheses	0.8	rdfs:label (or fragment)
defaultStopwords	0.7	rdfs:label (or fragment)
equality	0.6	+ skos:preflabel, skos:altLabel
normalize	0.5	+ skos:preflabel, skos:altLabel
normalizeParentheses	0.4	+ skos:preflabel, skos:altLabel
defaultStopwords	0.3	+ skos:preflabel, skos:altLabel

case there are multiple target entities for one source entity (or the other way around). Thus the following filters only add additional confidences (with the `addAdditionalConfidence` function of YAAA [5]) and do not yet remove any correspondences:

- Similar Neighbors Filter
- Cosine Similarity Filter
- Common Properties Filter
- Type Filter

All these filters are explained in the following. The similar neighbors filter uses the instance alignment (generated by the previous string processing step) to count for each instance correspondence how many resources or literals are shared between the two instances. Figure 2 shows an example where two neighbors are detected for correspondence <one:Harry_Potter, two:Harry_Potter> because the literal "blue" and the resource "Gryffindor" is shared. Note that the properties are not taken into account (which is done later by the common properties filter). Thus we do not need a mapping of property "eyeColor" to "eye". We further exclude the properties `rdfs:label` and `skos:altLabel` and all properties which have the same literal as those. This will not count the literals which just repeats the name of the resource with a different (maybe not matched) property like "name". Two literals are the same when their lowercased lexical value is equal. The additional confidence is the absolute amount of neighbors.

The cosine similarity filter compares text which is extracted from instances. It is generated by iterating over all literals and checking if the datatype of it is `xsd:string`,`rdf:langString` or if the literal has a language tag. All lexical representations of such literals are concatenated to generate a textual representation. These representations are then compared with a cosine similarity which is added to the correspondence.

The common properties filter checks for each instance correspondence the number of shared properties. This heavily relies on already matched schema because all properties with the same URI are excluded beforehand. Thus we only check if the instances share some matched properties regardless of their objects. The number of overlap is then added to the correspondence.

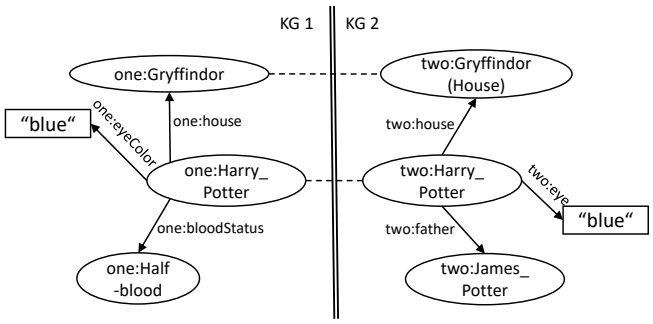


Fig. 2. The similar neighbors filter would assign two neighbors for the correspondence <one:Harry_Potter, two:Harry_Potter> because of literal "blue" and the already matched entites one:Gryffindor and two:Gryffindor(House).

The type filter is similar to the neighbors filter but only checks if the types (retrieved by `rdf:type`) actually overlap. This again requires already matched classes. The absolute overlap is added as an additional confidence.

The final step during instance matching is to actually filter these correspondences and create a one to one alignment. This instance filter sorts the correspondences by confidence (which is initially set by the string matching) and iterating over it. If a source or target resource is already matched, then it continues with the next correspondence. In all other cases it checks if there is a correspondence in the whole instance alignment which should be used instead. The criteria for being better is fixed to have greater values in two additional confidences.

As a last step, all correspondences are combined and a final cardinality filter ensures a one to one alignment by comparing the confidence scores.

1.2 Specific techniques used

We used the following matching components of MELT [5]:

- ScalableStringProcessingMatcher
- StopwordExtraction
- SimilarNeighborsFilter
- CommonPropertiesFilter
- CosineSimilarityConfidenceMatcher
- SimilarTypeFilter
- NaiveDescendingExtractor

1.3 Adaptations made for the evaluation

ATBox matcher is also available as a SEALS package. Due to clashes of dependecies of SEALS and ATBox, we decided to use the external SEALS packaging mechanism of the MELT framework[5]. It generates an intermediate matcher which executes an external process which runs in its own java virtual machine (JVM). Thus different versions of dependencies are not a problem.

1.4 Link to the system and parameters file

ATBox matcher can be downloaded from
https://www.dropbox.com/s/q57rzoec9zeumi2/ATBox.zip?dl=0.

2 Results

This section discusses the results of ATBox for each track of OAEI 2020 where the matcher is able to produce results. The following tracks are included: anatomy, conference, largebio, phenotype, and knowledge graph track.

Specific matching strategies and interfaces for the interactive and complex track are currently not implemented and are thus not described. Due to no multi language support, the multifarm track is also excluded.

2.1 Anatomy

ATBox could achive a slightly higher F-measure than the baseline (0.799 vs 0.766). Even though a synonym step is included in the matcher, the recall is only at 0.671 but therefor a high precision of 0.987 could be achieved (third best value).

Some examples were the matcher could find some non-trivial matches are:

- <cranium, Skull, =, 0.5>
- <lienal vein, Splenic_Vein, =, 0.5>
- <inner ear, Internal_Ear, =, 0.5>
- <celiac artery, Coeliac_Artery, =, 0.6>
- <grey matter, Gray_Matter, =, 0.6>

The first three have a confidence of 0.5 and thus the matches are mainly generated by synonym replacements. The last two contain different spellings like “grey” and “gray”. They are matched because the levenshtein distance is one between the two strings.

Some examples where the synonym step yields wrong results are:

- <naris, Nostril, =, 0.5>
- <upper arm, Biceps, =, 0.5>

This shows that not only true positives are generated and it is also the reason why the correspondence has a low confidence.

2.2 Conference

In the conference track ATBox matcher (0.56) is a bit better in terms of F-Measure than the baselines edna (0.54) and StringEquiv (0.52) when using the ra2-M3 evaluation. It covers the class and property alignments (M3) and uses the ra2 reference alignment which is a transitive closure of the original reference alignment ra1[9]. Analyzing the precision/recall triangular graph which is

based on the same evaluation dataset is can easily be seen that ATBox matcher has the best tradeoff between recall and precision. The reason is mainly the higher recall and the lower precision which is not easily avoidable. The schema matching capabilities of ATBox are rather limited and thus only the synonym expansion helps a lot. The ontology specific stopwords do not help here because they do not exist in the given dataset. Some examples where the synonym step help: <Trip, Excursion>, <Participant, Attendee>, <Place, Location>, and <SubjectArea, Topic>. The levenshtein distance helps finding <Sponsor, Sponzor> and <Organization, Organisation>. Furthermore ATBox is one of the seven matching systems which returns a wide variation of confidence values.

2.3 Largebio

ATBox matcher is one of six systems which are able to run on all six test cases and return meaningful alignments. It was consequently the second fastest system after LogMapLt. The results are very good in terms of precision but the recall is to low to compete with the other participants. Only in the FMA-SNOMED small fragments test cases the presented matcher could perform better than Wiktionary and LogMapLt.

2.4 Phenotype

In this track the presented matcher only returns 759 correspondences for the first task HP-MP and 1,318 correspondences for the second task DOID-ORDO. The evaluation result thus contains a low recall of 0.298 respectively 0.333. Together with a high precision, a F-measure of 0.457 and 0.498 can be achieved. This is probably due to the missing background knowledge because LogMapBio uses BioPortal, LogMap uses spelling variants of SPECIALIST lexicon, and AML uses three sources (Uberon, DOID, and MeSH). All these systems achieve a higher recall than ATBox. Nevertheless in task HP-MP we could rank higher than ALOD2Vec and Wiktionary.

2.5 Biodiv

In the Biodiv track ATBox could only return results in FLOPO-PTO test case. Once again the F-measure of 0.714 is much better than those of Wiktionary and ALOD2Vec but less than all LogMap variants and AML.

2.6 Knowledge Graph

ATBox could score int the overall evaluation(which contains classes, properties, and instances) the second highest F-measure score of 0.85 together with AML. Only ALOD2Vec and Wiktionary scores 0.01 better. When matching only classes, the presented matcher is the second best system after AML and for properties it is the best matcher. The instance matching pipeline is helpful for finding the correct correspondences but with 0.84 it is a bit below AML (0.85), ALOD2Vec (0.87), and Wiktionary (0.87).

3 General comments

3.1 Discussions on the way to improve the proposed system

We would like to increase the number of feature generators. For example, all texts connected to an instance could be compared not only with cosine similarity but also with a BERT classifier[2]. Another feature would be to compare images associated with the instances to further distinguish true positive from false positive correspondences.

Furthermore the schema matches could be improved with the help of all instance correspondences as already shown in DOME matcher [3].

4 Conclusions

In this paper, we have analyzed the results of ATBox matcher in OAEI 2020. It shows that the system is very scalable and can generate class, property and instance alignments. It usually has a high precision but on some tracks like Largebio, Phenotype, and Biodiv the recall can be increased by utilizing external knowledge despite the already used synonym lexicon from Wiktionary.

Most of the used matching components are furthermore included in the MELT framework[5] to allow other system developers to reuse them.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: Dbpedia: A nucleus for a web of open data. In: The semantic web, pp. 722–735. Springer (2007)
2. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
3. Hertling, S., Paulheim, H.: Dome results for oaei 2019. OM@ ISWC **2536**, 123–130 (2019)
4. Hertling, S., Paulheim, H.: The knowledge graph track at oaei - gold standards, baselines, and the golden hammer bias. In: The Semantic Web: ESWC 2020. pp. 343–359 (2020)
5. Hertling, S., Portisch, J., Paulheim, H.: Melt - matching evaluation toolkit. In: SEMANTICS. Karlsruhe. (2019)
6. Hoffart, J., Suchanek, F.M., Berberich, K., Weikum, G.: Yago2: A spatially and temporally enhanced knowledge base from wikipedia. Artificial Intelligence **194**, 28–61 (2013)
7. Levenshtein, V.I.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet physics doklady. vol. 10, pp. 707–710 (1966)
8. Sérasset, G.: Dbnary: Wiktionary as a lemon-based multilingual lexical resource in rdf. Semantic Web **6**(4), 355–361 (2015)
9. Zamazal, O., Svátek, V.: The ten-year ontofarm and its fertilization within the onto-sphere. Journal of Web Semantics **43**, 46–53 (2017)

Results of CANARD in OAEI 2020*

Elodie Thiéblin[1], Ollivier Haemmerlé[2], and Cassia Trojahn[2]

[1] Logilab, France
elodie.thieblin@logilab.fr
[2] IRIT & Université de Toulouse 2 Jean Jaurès, Toulouse, France
ollivier.haemmerle@irit.fr, cassia.trojahn@irit.fr

Abstract. This paper presents the results from the CANARD system in the OAEI 2020 campaign. CANARD is a system able to generate complex alignments. It is based on the notion of competency questions for alignment, as a way of expressing user needs. The system has participated in tracks where instances are available (Populated Conference, Populated Geolink, Populated Enslaved and Taxon datasets). This is the third participation of CANARD in the OAEI campaigns.

1 Presentation of the system

The CANARD (Complex Alignment Need and A-box based Relation Discovery) system [3,4] discovers complex correspondences between populated ontologies based on Competency Questions for Alignment (CQAs). CQAs represent the knowledge needs of a user and define the scope of the alignment. They are competency questions that need to be satisfied over two or more ontologies. Our approach takes as input a set of CQAs translated into SPARQL queries over the source ontology. The answer to each query is a set of instances retrieved from a knowledge base described by the source ontology. These instances are matched with those of a knowledge base described by the target ontology. The generation of the correspondence is performed by matching the subgraph from the source CQA to the lexically similar surroundings of the target instances.

The system source code and the configuration files are available at `https://framagit.org/IRIT_UT2J/ComplexAlignmentGenerator`.

1.1 Settings definition

Following the evaluation made in [3], the number of support instances was set to 2 instead of 10 last year to improve the runtime. The levenshtein similarity threshold was set to 0.4 like last year.

1.2 Adaptations made for the evaluation

Automatic generation of CQAs OAEI tracks do not cover CQAs i.e., the CQAs can not be given as input in the evaluation. We extended last year's query generator so that it can output binary queries. The query generator now produces three types of SPARQL queries: *Classes*, *Properties* and *Property-Value pairs*.

Classes For each *owl:Class* populated with at least one instance, a SPARQL query is created to retrieve all the instances of this class. If `<o1#class1>` is a populated class of the source ontology, the following query is created:
`SELECT DISTINCT ?x WHERE {?x a <o1#class1>.}`

Properties For each *owl:ObjectProperty* or *owl:Dataproperty* with at least one instantiation in the source knowledge base, a SPARQL query is created to retrieve all instantiations of this property. If `<o1#property1>` is an instantiated property of the source ontology, the following query is created:
`SELECT DISTINCT ?x ?y WHERE {?x <o1#property1> ?y.}`

Property-Value pairs Inspired by the approaches of [1,2,5], we create SPARQL queries of the form

- `SELECT DISTINCT ?x WHERE {?x <o1#property1> <o1#Value1>.}`
- `SELECT DISTINCT ?x WHERE {<o1#Value1> <o1#property1> ?x.}`
- `SELECT DISTINCT ?x WHERE {?x <o1#property1> "Value".}`

These property-value pairs are computed as follow: for each property (object or data property), the number of distinct object and subject values are retrieved. If the ratio of these two numbers is over a threshold (arbitrarily set to 30) and the smallest number is smaller than a threshold (arbitrarily set to 20), a query is created for each of the less than 20 values. For example, if the property `<o1#property1>` has 300 different subject values and 3 different object values (`"Value1"`, `"Value2"`, `"Value3"`), the ratio $|subject|/|object| = 300/3 > 30$ and $|object| = 3 < 20$. The 3 following queries are created as CQAs:

- `SELECT DISTINCT ?x WHERE {?x <o1#property1> "Value1".}`
- `SELECT DISTINCT ?x WHERE {?x <o1#property1> "Value2".}`
- `SELECT DISTINCT ?x WHERE {?x <o1#property1> "Value3".}`

The threshold on the smallest number ensures that the property-value pairs represent a category. The threshold on the ratio ensures that properties represent categories and not properties with few instantiations.

Implementation adaptations In the initial version of the system, Fuseki server endpoints are given as input. For the SEALS evaluation, we embedded a Fuseki server inside the matcher. The ontologies are downloaded from the SEALS repository, then uploaded in the embedded Fuseki server before the matching process can start. This downloading-uploading phase takes time, in particular when dealing with large files.

The CANARD system in the SEALS package is available at http://doi.org/10.6084/m9.figshare.7159760.v2. The generated alignments over the datasets in which CANARD performed are available at:

- **Populated Conference**: http://oaei.ontologymatching.org/2020/results/complex/popconf/results_conference.zip
- **Populated GeoLink**: http://oaei.ontologymatching.org/2020/results/complex/popgeolink/popgeolink_results_2020.zip
- **Populated Enslaved**: http://oaei.ontologymatching.org/2020/results/complex/popenslaved/popenslaved_results_2020.zip
- **Taxon**: http://oaei.ontologymatching.org/2020/results/complex/taxon/results_taxon.zip

2 Results

Please refer to http://oaei.ontologymatching.org/2020/results/complex for the results of CANARD in the OAEI 2020 campaign.

2.1 Populated Conference

Two datasets were used in the Populated Conference subtrack, one with more instances than the other. CANARD could perform all the matching tasks in the smaller dataset but timed out on 16 out of the 20 oriented pairs. This highlights one of CANARD's limitations : scalability.

For this reason, the coverage score is much lower on the large dataset than on the small one. While merging the results of all matchers by taking their best run (original, small or large dataset), CANARD obtains the best coverage score. It is the only evaluated matcher with a Coverage score higher than that of the reference simple alignment (ra1).

2.2 Populated Geolink

CANARD achieved the best relaxed-precision score (0.89) and the second best relaxed-recall score (0.54). This score however does not consider the semantics of the output correspondence. Most systems achieved a high relaxed precision score. Because of the automatic generation of CQAs, many correspondences of the form $\exists gbo{:}hasPlatformType.\{X\} \equiv gmo{:}Platform$, where X is a platform type were found.

2.3 Populated Enslaved

CANARD performed the lowest in this track out of the three evaluated complex matchers. On the enslaved-wikidata oriented pair of ontologies, CANARD found many instance links for each support answer. These links were found with literal comparison on two instances, a generic method which brings a lot of errors on

a dataset with many literal information (such as dates or values). In the case of binary CQAs, as CANARD tries to find a property path between each aligned entity, the runtime exploded and had to be stopped. This shows a major flaw in CANARD that should be fixed.

2.4 Taxon

CANARD has output much more correspondences than last year. A recurring pattern was found in the correspondences: an object property from Taxref or Agrovoc is aligned to a chain of *agronomicTaxon:hasHigherRank* and *agronomicTaxon:hasLowerRank* properties. This lowered the precision score in comparison with last year's. This can be explained by:

- Wrong instance linking based on label matching regardless of the language (e.g., a plant taxon matched to a habitat)
- The computation of all possible links between the two matched instances in the target knowledge-base
- If a path is found between two matched instances, it gets a default confidence value of 0.5. If a better path is found (a path with a lexical similarity to the source property), it gets a higher value and the default path are filtered. In this case, no better path was found so all correspondences were kept.

The recall score is also lower as last year's because the system was set to use only 2 support instances this year. As the instances are not homogeneously described in each dataset, more support instances mean more chances of finding one in the target dataset which instantiate the initial knowledge need. However, CANARD still achieves the best Coverage scores.

3 General comments

CANARD relies on common instances between the ontologies. It works best with aligned instances as it will try to find lexically similar entities otherwise. Hence, when such instances are not available, the approach is not able to generated complex correspondences. Furthermore, CANARD is need-oriented and requires a set competency questions to guide the matching process. Here, these "questions" have been automatically generated based on a set of patterns.

CANARD's runtime is extremely long. It depends (among other things) on the performance of the SPARQL endpoint it interrogates and the presence (or not) of equivalent links.

However, even with generated queries (instead of user input CQAs) it obtains some of the best coverage scores.

4 Conclusions

This paper presented the adapted version of the CANARD system and its preliminary results in the OAEI 2020 campaign. This year, we have been participated in Populated Conference, Populated GeoLink, Populated Enslaved and Taxon track, in which ontologies are populated with common instances.

References

1. Parundekar, R., Knoblock, C.A., Ambite, J.L.: Linking and building ontologies of linked data. In: ISWC. pp. 598–614. Springer (2010)
2. Parundekar, R., Knoblock, C.A., Ambite, J.L.: Discovering concept coverings in ontologies of linked data sources. In: ISWC. pp. 427–443. Springer (2012)
3. Thiéblin, É.: Automatic Generation of Complex Ontology Alignments. (Génération automatique d'alignements complexes d'ontologies). Ph.D. thesis, Paul Sabatier University, Toulouse, France (2019), https://tel.archives-ouvertes.fr/tel-02735724
4. Thiéblin, É., Haemmerlé, O., Trojahn, C.: Generating expressive correspondences: An approach based on user knowledge needs and a-box relation discovery. In: Pan, J.Z., Tamma, V.A.M., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., Kagal, L. (eds.) The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I. Lecture Notes in Computer Science, vol. 12506, pp. 565–583. Springer (2020), https://doi.org/10.1007/978-3-030-62419-4_32
5. Walshe, B., Brennan, R., O'Sullivan, D.: Bayes-recce: A bayesian model for detecting restriction class correspondences in linked open data knowledge bases. International Journal on Semantic Web and Information Systems 12(2), 25–52 (2016)

DESKMatcher

Michael Monych[2][0000−0002−3333−6307], Jan Portisch[1,2][0000−0001−5420−0663], Michael Hladik[2][0000−0002−2204−3138], and Heiko Paulheim[1][0000−0003−4386−8195]

[1] Data and Web Science Group, University of Mannheim, Germany
{jan, heiko}@informatik.uni-mannheim.de
[2] SAP SE Product Engineering Financial Services, Walldorf, Germany
{michael.monych, jan.portisch, michael.hladik}@sap.com

Abstract. This paper describes *DESKMatcher*, a label-based ontology matcher. It utilizes background knowledge from the financial services and enterprise domain to better find matches in these domains. The background knowledge utilized for the enterprise domain was in the form of documentation of terms used in SAP software (textual). Therefore, *Word2Vec* and *GloVe* were used for these corpora. The *Financial Industries Business Ontology* (*FIBO*) was used as more specific background knowledge for the financial services domain. Vector space embeddings for this corpus were trained using *RDF2Vec* and *KGloVe*. Individual matchers utilizing one set of embeddings (generated from a combination of method and corpus) are pipelined together after a string-based matchers, searching only for matches between entities that have not been assigned to a match in a previous step. Results on the *OAEI* tracks are expected to be sub-par, because low overlap between corpus and task vocabulary is expected.[3]

Keywords: Ontology Matching · Ontology Alignment · Domain Specific Background Knowledge

1 Presentation of the System

1.1 State, Purpose, General Statement

DESKMatcher (Enterprise *D*omain *S*pecific *K*nowledge Matcher) is an element-level, label-based matcher which utilizes vector space embeddings trained by applying multiple techniques on three background knowledge datasets specific to the enterprise and financial services domain, namely the *Financial Industry Business Ontology* (FIBO), the *SAP Glossary*, as well as *SAP Term*. The matcher was implemented for domain-specific matching in the financial services domain where classic schema matching problems are common and can be modelled as ontology matching problems [11].

However, in this paper we evaluate in how far the matcher generalizes to non-business/other domains. The matcher has not been adapted for other tasks.

1.2 Specific Techniques Used

The *DESKMatcher* system is implemented as a matching pipeline of subsequent matching steps using multiple domain-specific datasets that were embedded with *RDF2Vec* or *word2vec* depending on their inherent structure. In the following a quick introduction to the datasets used as well as to RDF2Vec will be given.

External Domain-Specific Datasets Below, we quickly introduce the sources of background knowledge that have been used:

1. The *Financial Services Business Ontology* (*FIBO*) [3] is used as the most specific source of background knowledge. It is an ontology specific to the financial services domain maintained by the EDM council, with the possibility for outside authors to contribute[4]. The *FIBO* version used contained roughly 88,000 triples with roughly 12,000 unique URIs.
2. The SAP Glossary is a textual corpus describing terms that are relevant for SAP's Enterprise Resource Planning (ERP) software. The resource is not available as ontology but instead in the form of a losely structured text corpus. The glossary was last released in 2017. The set contained definitions for roughly 48,000 terms using roughly 14,000 unique words.
3. The *SAP Term* is larger than the *SAP Glossary* but follows the same objective. It is frequently updated. The resource is not available as ontology but instead in the form of a losely structured text corpus. For this work we used the version as of March 2020. The set contained definitions for roughly 62,000 terms using roughly 16,000 unique words.

Embedding Approaches Used In *word2vec* [8] Mikolov et al. present two vector space embedding approaches for textual corpora: *Skip-Gram* (SG) and *continuous bag of words* (CBOW). Embeddings are generated by building a neural network that models randomly drawn context windows given a word (SG) and vice versa (CBOW). *RDF2Vec* [15] is an embedding approach for knowledge graphs that has already been used before in the area of ontology matching [13]. Random walks are generated starting at each node in the knowledge graph. The set of generated walks is then regarded as sentences and a *word2vec* algorithm is applied. Thereby, a vector is obtained for each node and for each edge (that appear in the random walks) in the knowledge graph.[5] *GloVe* [9] is another embedding approach for textual corpora presented 2014 by Pennington et al. Embeddings are generated based on co-occurence probabilities of words in the input corpus. *KGloVe* [2] is an approach to generate embeddings on knowledge graphs presented by Cochez et al. in 2017. Node "co-occurence probabilities" are approximated in a first step, by applying a version of the Bookmark Coloring Algorithm (BCA) [1]. The probabilities are then fed to the standard *GloVe* model, which yields embeddings for each node in the graph. Embeddings for

[4] see `https://github.com/edmcouncil/fibo/blob/master/CONTRIBUTING.md`

[5] More information about RDF2Vec and its application can be found online: `http://rdf2vec.org/`

FIBO were trained using the *jrdf2Vec*[6] [12] framework, as well as Cochez et al.'s implemenation of their own *KGloVe* [2]. *SAP Glossary* and *SAP Term* were embedded with *word2vec* (using the *gensim*[7] library [14]) and *GloVe* as made available by Penningotn et al[8].

Configuration of Embedding approaches Skip-gram was chosen over *CBOW*. This was based on Mikolov et al.'s results, that *Skip-gram* is better in semantic tasks [8, p. 7], which has also been indicated in [16, p. 4]. Generally, higher dimensions lead to higher performance, however the gain in performance per added dimension seems to greatly decrease after 200 dimensions, wherever dimensions are reported. Therefore the dimensions were fixed at 200[9]. Based on recommended parameter settings from previous work, the *window-size* was fixed to 5, negative sampling with 15 noise words and a smoothing exponent of 0.75 (as per Mikolov et al.'s recommendation in the original paper) was used. The *Skip-gram* embeddings were generated using the implementation in the *gensim* library [14].
The walks required for the *RDF2Vec* model were generated using *jRDF2Vec* [12], while the training of the actual embeddings was conducted using *gensim*'s *Skip-gram* implementation (same as for the text corpora). The walk strategy used to generate walks, is exactly one of the strategies proposed by Ristoski et al. in their original paper (Breadth-first [15]). 100 walks were generated per entity, using a depth of 4, which lead to "sentences" with a maximum length of 12.
To generate the *GloVe* embeddings, the original authors' `C` implementation was used [10]. For *GloVe* three parameters needed to be set: *minCount* was set to 4 in accordance to the value used in *Skip-gram*. *windowsize* was set to 15. x_{max} was set to 10 for this small corpus setting, due to the authors chosing 100 on their large corpus [9].
The implementation by Cochez et al. [11], was used to generate the shuffled co-occurrence files needed as input for the final step of *GloVe*.
Based on their results for best performance, the *PageRank* weighting scheme for context generation would have been chosen, which unfortunately did not execute without fatal errors, even after several attempts to tinker with the code. Therefore the uniform weighting was chosen, because it was reported to be the second best approach.
For the BCA, that is used to generate the "co-occurence probabilities", parameters α (which probability fraction is retained on a node) and ϵ (minimum value of probability to be distributed, values below being discarded) were chosen identical to the number Cochez et al. chose ($\alpha = 0.1$ and $\epsilon = 0.00001$).
The output co-occurence matrix was then put into *GloVe* using the same parameters as above.

[6] see `https://github.com/dwslab/jrdf2vec`
[7] see `https://radimrehurek.com/gensim/`
[8] see `https://nlp.stanford.edu/projects/glove/`
[9] In order to add another level of consistency between the approaches, the dimensions were also fixed to 200 in all of the other embedding generation approaches.
[10] available under `https://github.com/stanfordnlp/GloVe`
[11] `https://github.com/miselico/globalRDFEmbeddingsISWC`

Six embedding sets were therefore generated in total: two for each of the three corpora.

Matching Process Only the label and the entity type (class, datatype, property, object property, or individual) are considered. The entity types are used as a filter to only be matched against each other so that a homogeneous alignment is created, which proved to be a valuable heuristic in development. Matches are mainly determined based on the entity label. In the first step of the pipeline, simple matches are detected by a string matcher assuming n:m arity. Following steps try to apply increasingly less specific background knowledge in the form of embeddings trained on respectively less specific corpora, assuming only 1:1 arity (by ignoring entities already appearing in predicted matches). The specificity was assumed from the vocabulary size of a corpus. Per corpus, *Word2Vec/RDF2Vec* were applied before *GloVe/KGloVe* embeddings[12]. So the embedding sets were applied in the order *FIBO-RDF2Vec*, *FIBO-KGloVe*, *SAP Glossary-Word2Vec*, *SAP Glossary-Word2Vec*, *SAP Term-Word2Vec*, *SAP Term-GloVe*.

Implementation The system has been implemented and packaged with the *Matching and Evaluation Toolkit* (MELT), a framework for matcher development, tuning, evaluation, and packaging [4, 10]. As the matcher heavily depends on the python environment, the ML server module [5] of MELT has been forked to wrap additional python code. Eventually, the system was packaged with the framework. MELT greatly facilitated matcher development and also allowed for an easy inclusion of correspondence-level expanations.

2 Results

2.1 Anatomy

For this track, *DESKMatcher* was barely able to exceed the *StringEquiv* baseline and heavily underperformed on Precision and in turn F_1. Because the knowledge to train the embeddings was not taken from the same domain, these results are not surprising.

2.2 Conference

The Recall of 0.5 was rather below average compared to other matching systems, whereas Precision and F_1 were far below that of the others. An overlap between the Conference vocabulary present in the track and Business vocabulary from the background knowledge might have been expected, which in turn would have caused *DESKMatcher* to perform better.

[12] The decision whether to apply *Word2Vec/RDF2Vec* or *GloVe/KGloVe* embeddings first was taken arbitrarily. An improvement would be to investigate which embedding approaches actually are most suited for matching tasks.

2.3 Knowledge Graph

DESKMatcher was able to perform all test cases of the knowledge graph track [6]. In order to increase the performance, the embeddings are not used for instance matching. With an F_1 of 0.81, the matching system could outperform several systems on this track such as all 2020 *LogMap* [7] matching systems. Yet the F-score is still close to the `baselineLabel` matcher and below the `baselineAltLabel` matcher.

3 General Comments

3.1 Comments on the results (strength and weaknesses)

This system uses very specific domain knowledge from the financial services and business domains, which are not exactly covered by any of the tracks. Therefore, it was expected, that it should not be able to perform well. Even though expectations were set low, the results appear to be even worse. The system's strength lies in it being able to improve recall, which causes its greatest weakness: bad precision that in turn leads to bad F_1.

3.2 Discussions on the way to improve the proposed system

The greatest weakpoint of bad precision needs to be removed. Possible solutions would be a more strict linking process. A very greedy linking approach was chosen, to be able to find any matches at all. Additionally, the embedding sets can be pre-evaluated in a different way and discarded or used accordingly; using multiple embedding sets for one corpus did not show any positive results in the datasets evaluated here.

4 Conclusions

In this paper, we presented the *DESKMatcher*, a matching system for the financial services domain. The inner workings of the systems have been explained and the performance numbers in the 2020 campaign of the OAEI have been discussed. The system did not perform competitively in the campaign due to low vocabulary overlap in the datasets that have been used. We strive to improve the system in the future.

Bibliography

1. Berkhin, P.: Bookmark-coloring algorithm for personalized PageRank computing. Internet Mathematics 3(1), 41–62 (2006)
2. Cochez, M., Ristoski, P., Ponzetto, S.P., Paulheim, H.: Global RDF Vector Space Embeddings. In: d'Amato, C., Fernandez, M., Tamma, V., Lecue, F., Cudré-Mauroux, P., Sequeda, J., Lange, C., Heflin, J. (eds.) The Semantic Web – ISWC 2017, vol. 10587, pp. 190–207. Springer International Publishing, Cham (2017)
3. EDM Council: About FIBO. `https://edmcouncil.org/general/custom.asp?page=aboutfiboreview` (2019), (accessed 2020-09-02)
4. Hertling, S., Portisch, J., Paulheim, H.: MELT - Matching EvaLuation Toolkit. In: Semantics 2019 SEM2019 Proceedings. Karlsruhe (2019, to appear)
5. Hertling, S., Portisch, J., Paulheim, H.: Supervised ontology and instance matching with MELT. In: OM@ISWC 2020 (2020), to appear
6. Hofmann, A., Perchani, S., Portisch, J., Hertling, S., Paulheim, H.: Dbkwik: Towards knowledge graph creation from thousands of wikis. In: Nikitina, N., Song, D., Fokoue, A., Haase, P. (eds.) Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017. CEUR Workshop Proceedings, vol. 1963. CEUR-WS.org (2017), `http://ceur-ws.org/Vol-1963/paper540.pdf`
7. Jiménez-Ruiz, E.: Logmap family participation in the oaei 2020. OM@ISWC 2020 (2020), to appear
8. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space (Jan 2013)
9. Pennington, J., Socher, R., Manning, C.: Glove: Global Vectors for Word Representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 1532–1543. Association for Computational Linguistics, Doha, Qatar (2014)
10. Portisch, J., Hertling, S., Paulheim, H.: Visual analysis of ontology matching results with the melt dashboard. In: The Semantic Web: ESWC 2020 Satellite Events (2020)
11. Portisch, J., Hladik, M., Paulheim, H.: Evaluating Ontology Matchers on Real-World Financial Services Data Models p. 5 (2019)
12. Portisch, J., Hladik, M., Paulheim, H.: Rdf2vec light - A lightweight approach for knowledge graph embeddings. CoRR abs/2009.07659 (2020), `https://arxiv.org/abs/2009.07659`
13. Portisch, J., Paulheim, H.: ALOD2Vec Matcher. OM@ISWC. CEUR Workshop Proceedings (vol. 2288), pp. 132–137 (2018)
14. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks. pp. 45–50. ELRA, Valletta, Malta (May 2010)
15. Ristoski, P., Paulheim, H.: RDF2Vec: RDF Graph Embeddings for Data Mining. In: Groth, P., Simperl, E., Gray, A., Sabou, M., Krötzsch, M., Lecue, F., Flöck, F., Gil, Y. (eds.) The Semantic Web – ISWC 2016, vol. 9981, pp. 498–514. Springer International Publishing, Cham (2016)
16. Sheikh, I., Illina, I., Fohr, D., Linares, G.: Document Level Semantic Context for Retrieving OOV Proper Names. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 6050–6054. Proceeding of IEEE ICASSP 2016, IEEE, Shanghai, China (Mar 2016)

FTRLIM Results for OAEI 2020 ⋆ ⋆⋆

Xiaowen Wang[1], Yizhi Jiang[1], Hongfei Fan[1],
Hongming Zhu[1]⋆⋆⋆, and Qin Liu[1]

School of Software Engineering, Tongji University, Shanghai, China
{1931533,1931566,fanhongfei,zhu_hongming,qin.liu}@tongji.edu.cn

Abstract. FTRLIM is a distributed framework that is designed for large-scale instance matching. The FTRLIM framework leverages the blocking algorithm to generate candidate instance pairs, and applies the follow-the-regularized-leader model to determine whether candidate instance pairs are matched. FTRLIM participated in the SPIMBENCH Track of OAEI 2020, and achieved the fastest matching efficiency both in SANDBOX and MAINBOX, as well as the competitive matching quality.

1 Presentation of the system

1.1 State, purpose, general statement

The instance-based matching has gradually become a promising topic recently[1]. Many methods have been proposed to complete the instance matching task. Several state-of-the-art instance matching methods evolve from ontology matching methods such as LogMap[2], AML[3], RiMOM-IM[4], and Lily[5]. As the scale of the data increases, the efficiency and cost requirements of instance matching methods become more stringent.

FTRLIM is a distributed instance matching framework that focus more on the matching efficiency. When matching instances, it first generates indexes for instances based on their attributes. Instances with the same index are divided into the same instance block, and instances from different sources under the same block form the candidate instance pairs. Then FTRLIM figures out the matched instance pairs leveraging the online-learning model, follow-the-regularized-leader (FTRL). This is the second time that FTRLIM has participated in the OAEI evaluation. To participate in the SPIMBENCH Track, FTRLIM is rebuilt using JAVA with core functionalities as the submitted version. The complete version of FTRLIM has been developed and deployed on a Spark cluster, which provides the FTRLIM framework with ability to deal with large-scale data. Compared

⋆⋆ This research has been supported by the National Key R&D Program of China (No. 2018YFB0505000), the National Natural Science Foundation of China (No. 61702374), and the Fundamental Research Funds for the Central Universities.

⋆⋆⋆ Corresponding author, email: zhu_hongming@tongji.edu.cn

with last year's version, this year's FTRLIM has been slightly changed, which will be introduced later.

1.2 Specific techniques used

This section introduces the refined working flow of FTRLIM. FTRLIM consists of four major components: *Blocker*, *Comparator*, *Trainer*, and *Matcher*. The framework accepts input instances in the OWL format, which are stored in source dataset and target dataset, respectively. FTRLIM finds matched instances between the two datasets. The overview of the FTRLIM's work flow is presented in Fig. 1.

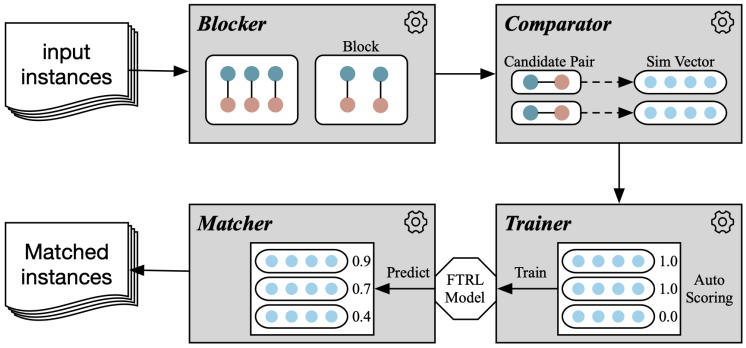


Fig. 1. Work Flow of FTRLIM OAEI 2020

Blocker Since the scale of instances that need to be matched is usually very large, it is very time-consuming and space-consuming to compare all the instances with each other to find matched instance pairs. *Blocker* extracts features of textual attributes related to instances to generate indexes for them. The interactions among different textual information are taken into consideration, which allows instances to be fine-grained divided. It also has the ability to infer indexes for instances whose textural attributes are in-completed or missing. FTRLIM supports users to generate indexes for instances via more than one attribute. Instances with the same index are divided into the same instance block, and instances from different sources under the same block will form candidate instance pairs. Only when a pair of instances is a candidate pair can it be matched in the following procedures. When there are only two instances from different data sources in the same block, these two instances will form a unique instance pair[4], which will be regarded as an matched instance pair directly.

Comparator All candidate pairs will be sent to the comparator to calculate similarity. The comparator compares two instances from user-specified aspects.

The edit distance similarity is calculated for textual instance attributes, while the Jaccard similarity is calculated for instance relationships. The calculation results will be arranged in order to form the similarity vector. Formally, let the list of predicates adopted by *Comparator* be $\langle p_1, p_2, \ldots, p_n \rangle$, then the similarity vector of the two instance is

$$\langle s_1, s_2, \ldots, s_n \rangle, s_i \in [0, 1], (i = 1, 2, \ldots, n)$$

where s_i is the similarity of the two instances under the i-th predicate.

Trainer FTRLIM treats the instance matching as a regression problem, where the similarity score between two instances can be regarded as the probability that the two instances are matched. We innovatively introduce the FTRL model[6] to solve the problem. FTRL is a widely-used online logistic regression model with high precision, excellent sparsity, fast training speed and satisfactory streaming data processing ability. *Trainer* is designed to train the FTRL model for instance matching. It first generates train set for the FTRL model. After the preparation of train set is completed, the FTRL model will be trained with hyperparameters in configuration files. Benefiting from the FTRL model's feature, the training process won't cost a long time. The *Trainer* component plays a greater role in the complete version. It can be used to accept the feedback of users and adjust the parameters of the FTRL model. Users are allowed to choose a batch of candidate instance pairs and correct the similarity score, or pick up a certain pair to correct.

Matcher All candidate pairs will obtain their final similarity scores in this component. Since FTRLIM produces the similarity scores in the interval [0,1], candidate pairs whose scores are greater than 0.5 will be regarded as matched pairs. The matching score s is calculated as follows:

$$s = \frac{1}{1 + e^{-\boldsymbol{x}^T \boldsymbol{w}}} \tag{1}$$

where $\boldsymbol{x}$ is the similarity vector, $\boldsymbol{w}$ is the weight of the FTRL model. In this year's submission, all elements of similarity vectors accepted by the FTRL model are unified from [0, 1] to [-1, 1] to satisfy the symmetry of the equation.

Configurations FTRLIM is easily to be tailored according to user's requirements. We expect that all matching procedures are under user's control, thus we allow users to customize their own FTRLIM system using configuration files. Users are able to set the attributes for index generation, the attributes and relationships for comparison, the hyperparameters for the FTRL model and many other detailed parameters to get a better result.

1.3 Adaptions made for the evaluation

To participate in the evaluation, we rebuilt FTRLIM and replaced some manual operations with automatic strategies.

The train set for training the FTRL model is automatically generated in the submitted version, while it needs manual scoring in the completed version. The train set is composed of instance pairs' similarity vectors as well as their similarity scores. The *Trainer* regards all unique pairs as matched pairs. Therefore, it selects all similarity vectors of unique pairs as positive samples, and assigns them with similarity score 1.0. The mismatched pairs are built by replacing one instance of each unique pair randomly. These pairs are assigned with similarity score 0.0 and treated as negative samples in the train set. In the completed version, however, FTRLIM does not regard all unique pairs as matched pairs directly. It will compute the mean value of similarity vectors' elements as the raw score for each instance pairs. Then it will select a batch of instance pairs that have raw scores higher than a threshold as positive samples, as well as the same amount of instance pairs whose raw scores are lower than the threshold as negative samples. Users will determine the similarity score by themselves to generate the train set. Besides, we excluded the non-core functionalities of FTRLIM such as the user-feedback and the load balance mechanism. The ways of input and output is adapted for the evaluation as well.

1.4 Link to the system and parameters file

The implementation of FTRLIM and relevant System Adapter for HOBBIT platform can be found at this FTRLIM-HOBBIT's gitlab page.[1]

2 Result

In this section, we present the results obtained by FTRLIM in the OAEI 2020 competition. FTRLIM participated in the SPIMBENCH Track, which aims at determining whether two OWL instances describe the same Creative Work. The datasets are generated and transformed using SPIMBENCH[7]. Our competitors includes LogMap[2], AML[3], Lily[5] and REMinder. The first three systems have participated in this track for many years, while REMinder is a new contestants in this year. The results are published in this OAEI 2020 result page[2].

2.1 SPIMBENCH

The SPIMBENCH task is executed in two datasets, the SANDBOX and the MAINBOX, of different size. The SANDBOX has about 380 instances and 10000 triplets, while the MAINBOX has about 1800 instances and 50000 triplets. We summarized the results of the SPIMBENCH Track in Table 1 and Table 2, where the best results are indicated in bold.

Table 1. The Results of SANDBOX

	LogMap	AML	Lily	FTRLIM	REMiner
Fmeasure	0.8413	0.8645	0.9917	0.9214	**0.9983**
Precision	0.9383	0.8349	0.9836	0.8542	**1**
Recall	0.7625	0.8963	**1**	**1**	0.9967
Time performance	7483	6446	2050	**1525**	7284

Table 2. The Results of MAINBOX

	LogMap	AML	Lily	FTRLIM	REMiner
Fmeasure	0.7856	0.8605	0.0.9954	0.9215	**0.9977**
Precision	0.8801	0.8385	0.9908	0.8558	**0.9987**
Recall	0.7095	0.8835	**1**	0.9980	0.9967
Time performance	26782	38772	3899	**2247**	33966

Compared with all competitors, FTRLIM achieves the best time performance on both two datasets. The time cost of our framework is reduced by 25.6% than the second fastest one, Lily, on SANDBOX, while it is reduced by 42.4% than Lily on MAINBOX. The results on time performance indicate the efficiency of FTRLIM, which is more essential for large-scale instance matching. The FTRLIM also achieves the highest recall on SANDBOX and almost the highest recall on MAINBOX. The precision of FTRLIM is relatively low on both datasets. There are two reasons that account for this situation. One reason is that the automatic strategy we adopted for generating train set is flawed. In the generated train set, there is almost no similarity between the the sample instance pairs with low score. Although this kind of samples helps the FTRL model learn to distinguish similar instance pairs from dissimilar instance pairs, it does not help the model distinguish matched instance pairs from similar instance pairs. Then the model prefers to predict high similarity scores for similar instance pairs, which improves the recall but reduces the precision. Another reason is that there may be problems with the way unique pairs are treated. Regarding the unique pairs as matched pairs directly will also affect the precision of the prediction. But the overall matching quality of FTRLIM is still competitive.

3 General comments

3.1 Comments on the result

FTRLIM has achieved time performance in both datasets of SPIMBENCH. The *Blocker* component makes a significant contribution to achieving the results. It helps the framework filter out instance pairs with a high possibility to be

[1] https://git.project-hobbit.eu/937522035/ftrlimhobbit
[2] http://oaei.ontologymatching.org/2020/results

matched effectively and efficiently. The *Comparator* component only needs to compare instances with the same indexes rather than every instance pairs. The datasets of SPIMBENCH contain a wealth of textual information, and there are many attributes that can be used to build indexes or to compare the similarity among instances. The FTRL model trained by *Trainer* is able to learn a weight for attributes or relationships and distinguish instance pairs that points to the same entity in the real world. Compared with other systems, the precision of FTRLIM is unsatisfactory, which should be improved in feature works.

3.2 Improvements

There are still many aspects to be improved in FTRLIM. The submitted version of FTRLIM generates flawed train set for training the FTRL model, and considers unique pairs as matched instances unconditionally. The automatic strategy adopted by *Trainer* and *Matcher* should be optimized to address the problems. More comparison methods for various data types should be attached to our frameworks as well. Although FTRLIM is specially designed to solve the instance matching problem, it is also expected to produce meaningful results in other similar tracks in the future.

4 Conclusion

In this paper, we briefly presented our instance matching framework FTRLIM. The core functionalities and components of FTRLIM were introduced, and the evaluation results of FTRLIM were presented and analyzed. FTRLIM achieved significantly better time performance than other systems on both two datasets of SPIMBENCH, as well as the competitive matching quality. The results indicated the effectiveness and high efficiency of our matching strategy, which is important for matching instances on large-scale datasets.

References

1. Otero-Cerdeira, L., Rodríguez-Martínez, F.J., Gómez-Rodríguez, A.: Ontology matching: A literature review. Expert Systems with Applications **42**(2), 949–971 (2015)
2. Jiménez-Ruiz, E., Grau, B.C.: Logmap: Logic-based and scalable ontology matching. In: International Semantic Web Conference. pp. 273–288. Springer (2011)
3. Faria, D., Pesquita, C., Santos, E., Cruz, I.F., Couto, F.M.: Agreementmakerlight results for oaei 2013. In: OM. pp. 101–108 (2013)
4. Shao, C., Hu, L., Li, J.Z., Wang, Z., Chung, T.L., Xia, J.B.: Rimom-im: A novel iterative framework for instance matching. Journal of Computer Science and Technology **31**, 185–197 (2016)
5. Wu, J., Pan, Z., Zhang, C., Wang, P.: Lily results for oaei 2019. In: OM@ ISWC. pp. 153–159 (2019)

6. McMahan, H.B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., et al.: Ad click prediction: a view from the trenches. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 1222–1230 (2013)
7. Saveta, T., Daskalaki, E., Flouris, G., Fundulaki, I., Herschel, M., Ngomo, A.C.N.: Spimbench : A scalable , schema-aware instance matching benchmark for the semantic publishing domain (2014)

Lily Results for OAEI 2020* **

Yunyan Hu [1], Shaochen Bai [2], Shiyi Zou [4], Peng Wang [1,2,3,4] ***

[1] School of Computer Science and Engineering, Southeast University, China
[2] School of Artificial Intelligence, Southeast University, China
[3] School of Cyber Science and Engineering, Southeast University, China
[4] Southeast University - Monash University Joint Graduate School
{yunyhu, baisc, shiyizou, pwang} @ seu.edu.cn

Abstract. This paper presents the results of Lily in the ontology alignment contest OAEI 2020. As a comprehensive ontology matching system, Lily is intended to participate in three tracks of the contest: anatomy, conference, and spimbench. The specific techniques used by Lily will be introduced briefly. The strengths and weaknesses of Lily will also be discussed.

1 Presentation of the system

With the use of hybrid matching strategies, Lily, as an ontology matching system, is capable of solving some issues related to heterogeneous ontologies. It can process normal ontologies, weak informative ontologies [1], ontology mapping debugging [2], and ontology matching tunning [3], in both normal and large scales. In previous OAEI contests [4–11], Lily has achieved preferable performances in some tasks, which indicated its effectiveness and wideness of availability.

1.1 State, purpose, general statement

The core principle of matching strategies of Lily is utilizing the useful clues correctly and effectively. Lily combines several effective and efficient matching techniques to facilitate alignments. There are five main matching strategies: (1) Generic Ontology Matching (GOM) is used for common matching tasks with normal size ontologies. (2) Large scale Ontology Matching (LOM) is used for the matching tasks with large size ontologies. (3) Instance Ontology Matching (IOM) is used for instance matching tasks. (4) Ontology mapping debugging is used to verify and improve the alignment results. (5) Ontology matching tuning is used to enhance overall performance.

** This work is supported by National Key R&D Program of China (2018YFD1100302) and 13th Five-Year All-Army Common Information System Equipment Pre-Research Project (No.31511110310).

*** Corresponding author pwang@seu.edu.cn (Peng Wang)

The matching process mainly contains three steps: (1) Pre-processing, when Lily parses ontologies and prepares the necessary information for subsequent steps. Meanwhile, the ontologies will be generally analyzed, whose characteristics, along with studied datasets, will be utilized to determine parameters and strategies. (2) Similarity computing, when Lily uses special methods to calculate the similarities between elements from different ontologies. (3) Post-processing, when alignments are extracted and refined by mapping debugging.

In this year, some algorithms and matching strategies of Lily have been modified for higher efficiency.

1.2 Specific techniques used

Lily aims to provide high quality 1:1 concept pair or property pair alignments. The main specific techniques used by Lily are as follows.

Semantic subgraph An element may have heterogeneous semantic interpretations in different ontologies. Therefore, understanding the real local meanings of elements is very useful for similarity computation, which are the foundations for many applications including ontology matching. Therefore, before similarity computation, Lily first describes the meaning for each entity accurately. However, since different ontologies have different preferences to describe their elements, obtaining the semantic context of an element is an open problem. The semantic subgraph was proposed to capture the real meanings of ontology elements [12]. To extract the semantic subgraphs, a hybrid ontology graph is used to represent the semantic relations between elements. An extracting algorithm based on an electrical circuit model is then used with new conductivity calculation rules to improve the quality of the semantic subgraphs. It has been shown that the semantic subgraphs can properly capture the local meanings of elements [12].

Based on the extracted semantic subgraphs, more credible matching clues can be discovered, which help reduce the negative effects of the matching uncertainty.

Generic ontology matching method The similarity computation is based on the semantic subgraphs, which means all the information used in the similarity computation comes from the semantic subgraphs. Lily combines the text matching and structure matching techniques.

Semantic Description Document (SDD) matcher measures the literal similarity between ontologies. A semantic description document of a concept contains the information about class hierarchies, related properties and instances, and external knowledge sources. A semantic description document of a property contains the clues about hierarchies, domains, ranges, restrictions and related instances. In addition, WordNet [13] and domain-specific ontologies (the UBERON [14] Ontology for the Anatomy track) are exploited as external resources to find synonyms and cross-references between entities. Indeed, we explore the property "hasDbXref", which is mentioned in almost every class of Uberon. This property references the classes'URI of some external ontologies

such as the human and mouse of the Anatomy track. Consequently, we align every two entities of the Anatomy track in case if they are both referenced in a single class of Uberon.

For the descriptions from different entities, the similarities of the corresponding parts will be calculated. Finally, all separated similarities will be combined with the experiential weights.

Matching weak informative ontologies Most existing ontology matching methods are based on the linguistic information. However, some ontologies may lack in regular linguistic information such as natural words and comments. Consequently the linguistic-based methods will not work. Structure-based methods are more practical for such situations. Similarity propagation is a feasible idea to realize the structure-based matching. But traditional propagation strategies do not take into consideration the ontology features and will be faced with effectiveness and performance problems. Having analyzed the classical similarity propagation algorithm, *Similarity Flood*, we proposed a new structure-based ontology matching method [1]. This method has two features: (1) It has more strict but reasonable propagation conditions which lead to more efficient matching processes and better alignments. (2) A series of propagation strategies are used to improve the matching quality. We have demonstrated that this method performs well on the OAEI benchmark dataset [1].

However, the similarity propagation is not always perfect. When more alignments are discovered, more incorrect alignments would also be introduced by the similarity propagation. So Lily also utilizes a strategy to determine when to use the similarity propagation.

Large scale ontology matching Matching large ontologies is a challenge due to its significant time complexity. We proposed a new matching method for large ontologies based on reduction anchors [15]. This method has a distinct advantage over the divide-and-conquer methods because it does not need to partition large ontologies. In particular, two kinds of reduction anchors, positive and negative reduction anchors, are proposed to reduce the time complexity in matching. Positive reduction anchors use the concept hierarchy to predict the ignorable similarity calculations. Negative reduction anchors use the locality of matching to predict the ignorable similarity calculations. Our experimental results on the real world datasets show that the proposed methods are efficient in matching large ontologies [15].

Ontology mapping debugging Lily utilizes a technique named *ontology mapping debugging* to improve the alignment results [2]. Different from existing methods that focus on finding efficient and effective solutions for the ontology mapping problems, mapping debugging emphasizes on analyzing the mapping results to detect or diagnose the mapping defects. During debugging, some types of mapping errors, such as redundant and inconsistent mappings, can be detected. Some

warnings, including imprecise mappings or abnormal mappings, are also locked by analyzing the features of mapping result. More importantly, some errors and warnings can be repaired automatically or can be presented to users with revising suggestions.

Ontology matching tuning Lily adopted ontology matching tuning this year. By performing parameter optimization on training datasets [3], Lily is able to determine the best parameters for similar tasks. Those data will be stored. When it comes to real matching tasks, Lily will perform statistical calculations on the new ontologies to acquire their features that help it find the most suitable configurations, based on previous training data. In this way, the overall performance can be improved.

Currently, ontology matching tuning is not totally automatic. It is difficult to find out typical statistical parameters that distinguish each task from others. Meanwhile, learning from test datasets can be really time-consuming. Our experiment is just a beginning.

2 Results

2.1 Anatomy track

The anatomy matching task consists of two real large-scale biological ontologies. Table 1 shows the performance of Lily in the Anatomy track on a server with one 3.46 GHz, 6-core CPU and 8GB RAM allocated. The time unit is second (s).

Table 1. The performance in the Anatomy track

Matcher	Runtime	Precision	Recall	Recall+	F-Measure
Lily	706	0.901	0.902	0.747	0.901

Compared with the result in OAEI 2019 [5], there are some improvements in Precision, Recall and F-Measure. However, as can be seen in the overall result, there are still some gaps compared with the state-of-art system which indicates it is still possible to make further progress. The further exploration of external knowledge will be leveraged in the future for the better results. Additionally, to futher reduce the time consumption, some key algorithms will be parallelized.

2.2 Conference track

In this track, there are 7 independent ontologies that can be matched with one another. The 21 subtasks are based on given reference alignments. As a result of

heterogeneous characters, it is a challenge to generate high-quality alignments for all ontology pairs in this track.

Lily adopted ontology matching tuning for the Conference track this year. Table 2 shows its latest performance.

Table 2. The performance in the Conference track

Test Case ID	Precision	Recall	F.5-Measure	F1-measure	F2-measure
ra1-M1	0.67	0.57	0.65	0.62	0.59
ra1-M3	0.67	0.47	0.62	0.55	0.5
ra2-M1	0.67	0.49	0.62	0.57	0.52
ra2-M3	0.63	0.42	0.63	0.57	0.50
rar2-M1	0.62	0.52	0.6	0.57	0.54
rar2-M3	0.62	0.43	0.57	0.51	0.46
Average	**0.65**	**0.48**	**0.62**	**0.57**	**0.52**

Compared with the result in OAEI 2018 [5], there is one very slightly improved its precision but decreased its recall and F1-measure. All the tasks share the same configurations, so it is possible to generate better alignments by assigning the most suitable parameters for each task. The performance of Lily was even worse than StringEquiv in some tasks. 'We will further analyze this task and our system to find out the reason later.

2.3 Spimbench track

This tack is an instance-mactching tack which aims to match instances of creative works between two boxes. And ontology instances are described through 22 classes, 31 DatatypeProperty and 85 ObjectProperty properties.

There are about 380 instances and 10000 triples in sandbox, and about 1800 CWs and 50000 triples in mainbox.

Table 3. Performance in the spimbench task

Track	Matcher	Precision	Recall	F-Measure	Time
SANDBOX	AML	0.8349	0.8963	0.8645	6446
	FTRLIM	0.8543	**1.000**	0.9214	**1525**
	LogMap	0.9383	0.7625	0.8413	7483
	REMiner	**1.0**	0.9967	**0.9983**	7284
	Lily	0.9836	**1.000**	0.9917	2050
MAINBOX	AML	0.8386	0.8835	0.8605	38772
	FTRL-IM	0.8558	0.9980	0.9215	**2247**
	LogMap	0.8801	0.7095	0.7856	26782
	REMiner	**0.9986**	0.9966	**0.9977**	33966
	Lily	0.9908	**1.000**	0.9954	3899

Lily utilized almost the same startegy to handle these two different size tasks. We found that creative works in this task was rich in text information such as titles, descriptions and so on. However, garbled texts and messy codes were mixed up with normal texts. And Lily relied too much on text similarity calculation and set a low threshold in this task, which accounted for the low precision.

As is shown in Table 3, Lily outperforms most the others in sandbox and mainbox. And the results of Lily and REMiner are close, but the running time of Lily is comparative. Meanwhile, experiments shows that simple ensemble methods and a low threshold contribute to increase of matching efficiency. Nevertheless, compared with FTRL-IM, there is still potential for Lily to speed up in process of matching.

3 General comments

In this year, some modifications were done to Lily for both effectiveness and efficiency. The performance has been improved as we have expected. The strategies for new tasks have been proved to be useful.

On the whole, Lily is a comprehensive ontology matching system with the ability to handle multiple types of ontology matching tasks, of which the results are generally competitive. However, Lily still lacks in strategies for some newly developed matching tasks. The relatively high time and memory consumption also prevent Lily from finishing some challenging tasks.

4 Conclusion

In this paper, we briefly introduced our ontology matching system Lily. The matching process and the special techniques used by Lily were presented, and the alignment results were carefully analyzed.

There is still so much to do to make further progress. Lily needs more optimization to handle large ontologies with limited time and memory. Thus, techniques like parallelization will be applied more. Also, we have just tried out ontology matching tuning. With further research on that, Lily will not only produce better alignments for tracks it was intended for, but also be able to participate in the interactive track.

References

1. Wang, P., Xu, B.: An effective similarity propagation model for matching ontologies without sufficient or regular linguistic information. In: The 4th Asian Semantic Web Conference (ASWC2009), Shanghai, China (2009)
2. Wang, P., Xu, B.: Debugging ontology mappings: a static approach. Computing and Informatics **27**(1), 21–36 (2012)
3. Yang, P., Wang, P., Ji, L., Chen, X., Huang, K., Yu, B.: Ontology matching tuning based on particle swarm optimization: Preliminary results. In: Chinese Semantic Web and Web Science Conference. pp. 146–155. Springer (2014)

4. Wu, J., Pan, Z., Zhang, C., Wang, P.: Lily results for OAEI 2019. In: Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019. vol. 2536, pp. 153–159. CEUR-WS.org (2019)
5. Tang, Y., Wang, P., Pan, Z., Liu, H.: Lily results for OAEI 2018. In: Proceedings of the 13th International Workshop on Ontology Matching co-located with the 17th International Semantic Web Conference, OM@ISWC 2018, Monterey, CA, USA, October 8, 2018. pp. 179–186 (2018)
6. Wang, P., Wang, W.: Lily results for OAEI 2016. In: Proceedings of the 11th International Workshop on Ontology Matching co-located with the 15th International Semantic Web Conference (ISWC 2016), Kobe, Japan, October 18, 2016. pp. 178–184 (2016)
7. Wang, W., Wang, P.: Lily results for OAEI 2015. In: Proceedings of the 10th International Workshop on Ontology Matching collocated with the 14th International Semantic Web Conference (ISWC 2015), Bethlehem, PA, USA, October 12, 2015. pp. 162–170 (2015)
8. Wang, P.: Lily results on SEALS platform for OAEI 2011. In: Proceedings of the 6th International Workshop on Ontology Matching, Bonn, Germany, October 24, 2011 (2011)
9. Wang, P., Xu, B.: Lily: Ontology alignment results for OAEI 2009. In: Proceedings of the 4th International Workshop on Ontology Matching (OM-2009) collocated with the 8th International Semantic Web Conference (ISWC-2009) Chantilly, USA, October 25, 2009 (2009)
10. Wang, P., Xu, B.: Lily: Ontology alignment results for OAEI 2008. In: Proceedings of the 3rd International Workshop on Ontology Matching (OM-2008) Collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26, 2008 (2008)
11. Wang, P., Xu, B.: LILY: the results for the ontology alignment contest OAEI 2007. In: Proceedings of the 2nd International Workshop on Ontology Matching (OM-2007) Collocated with the 6th International Semantic Web Conference (ISWC-2007) and the 2nd Asian Semantic Web Conference (ASWC-2007), Busan, Korea, November 11, 2007 (2007)
12. Wang, P., Xu, B., Zhou, Y.: Extracting semantic subgraphs to capture the real meanings of ontology elements. Tsinghua Science and Technology **15**(6), 724–733 (2010)
13. Miller, G.A.: Wordnet: A lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
14. Mungall, C.J., Torniai, C., Gkoutos, G.V., et al.: Uberon, an integrative multi-species anatomy ontology. Genome biology **13**(1), R5 (2012)
15. Wang, P., Zhou, Y., Xu, B.: Matching large ontologies based on reduction anchors. In: IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011. pp. 2343–2348 (2011)

LogMap Family Participation in the OAEI 2020 *

Ernesto Jiménez-Ruiz[1,2]

[1] Department of Computer Science, City, University of London, UK
[2] Department of Informatics, University of Oslo, Oslo, Norway

Abstract. We present the participation of LogMap and its variants in the OAEI 2020 campaign. The LogMap project started in January 2011 with the objective of developing a scalable and logic-based ontology matching system. This is the ninth participation in the OAEI and the experience has so far been very positive. LogMap is one of the few systems that participates in (almost) all OAEI tracks.

1 Presentation of the system

LogMap [7, 9] is a highly scalable ontology matching system that implements the consistency and locality principles [8]. LogMap is one of the few ontology matching system that *(i)* can efficiently match semantically rich ontologies containing tens (and even hundreds) of thousands of classes, *(ii)* incorporates sophisticated reasoning and repair techniques to minimise the number of logical inconsistencies, and *(iii)* provides support for user intervention during the matching process.

1.1 LogMap variants in the 2020 campaign

As in previous campaigns, in the OAEI 2020 we have participated with two additional variants:

LogMapLt is a "lightweight" variant of LogMap, which essentially only applies (efficient) string matching techniques.

LogMapBio includes an extension to use BioPortal [4, 5] as a (dynamic) provider of mediating ontologies instead of relying on a few preselected ontologies [1].

In previous years we also participated with LogMapC[3].

1.2 Link to the system and parameters file

LogMap is open-source and released under GNU Lesser General Public License 3.0.[4] LogMap components and source code are available from the LogMap's GitHub page: `https://github.com/ernestojimenezruiz/logmap-matcher/`.

[3] LogMapC is a variant of LogMap which, in addition to the consistency and locality principles, also implements the conservativity principle (see details in [11]).

[4] `http://www.gnu.org/licenses/`

LogMap distributions can be easily customized through a configuration file containing the matching parameters.

LogMap, including support for interactive ontology matching, can also be used directly through an AJAX-based Web interface: `http://krrwebtools.cs.ox.ac.uk/`. This interface has been very well received by the community since it was deployed in 2012. More than 4,500 requests coming from a broad range of users have been processed so far.

1.3 LogMap as a mapping repair system

Only a very few systems participating in the OAEI competition implement repair techniques. As a result, existing matching systems (even those that typically achieve very high precision scores) compute mappings that lead in many cases to a large number of unsatisfiable classes.

We believe that these systems could significantly improve their output if they were to implement repair techniques similar to those available in LogMap. Therefore, with the goal of providing a useful service to the community, we have made LogMap's ontology repair module (LogMap-Repair) available as a self-contained software component that can be seamlessly integrated in most existing ontology matching systems [10, 3].

1.4 LogMap as a matching task division system

LogMap also includes a novel module to divide the ontology alignment task into (independent) manageable subtasks [6]. This component relies on LogMap's lexical index, a neural embedding model [12] and locality-based modules [2]. This module can be integrated in existing ontology alignment systems as a external module. The results in [6] are encouraging as the division enabled systems to complete some large-scale matching tasks.

2 General comments and conclusions

Please refer to `http://oaei.ontologymatching.org/2020/results/` for the results of the LogMap family in the OAEI 2020 campaign.

2.1 Comments on the results

As in previous campaigns, LogMap has been one of the top systems and one of the few systems that participates in (almost) all tracks. Furthermore, it has also been one of the few systems implementing repair techniques and providing (almost) coherent mappings in all tracks.

LogMap's main weakness is that the computation of candidate mappings is based on the similarities between the vocabularies of the input ontologies; hence, in the cases where the ontologies are lexically disparate or do not provide enough lexical information LogMap is at a disadvantage.

Acknowledgements

I would also like to thank Bernardo Cuenca-Grau, Ian Horrocks, Alessandro Solimando, Valerie Cross, Anton Morant, Yujiao Zhou, Weiguo Xia, Xi Chen, Yuan Gong and Shuo Zhang, who have contributed to the LogMap project in the past.

References

1. Chen, X., Xia, W., Jiménez-Ruiz, E., Cross, V.: Extending an ontology alignment system with bioportal: a preliminary analysis. In: Poster at Int'l Sem. Web Conf. (ISWC) (2014)
2. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U.: Modular reuse of ontologies: Theory and practice. J. Artif. Intell. Res. 31, 273–318 (2008)
3. Faria, D., Jiménez-Ruiz, E., Pesquita, C., Santos, E., Couto, F.M.: Towards annotating potential incoherences in bioportal mappings. In: 13th Int'l Sem. Web Conf. (ISWC) (2014)
4. Fridman Noy, N., Shah, N.H., Whetzel, P.L., Dai, B., et al.: BioPortal: ontologies and integrated data resources at the click of a mouse. Nucleic Acids Research 37, 170–173 (2009)
5. Ghazvinian, A., Noy, N.F., Jonquet, C., Shah, N.H., Musen, M.A.: What four million mappings can tell you about two hundred ontologies. In: Int'l Sem. Web Conf. (ISWC) (2009)
6. Jiménez-Ruiz, E., Agibetov, A., Chen, J., Samwald, M., Cross, V.: Dividing the Ontology Alignment Task with Semantic Embeddings and Logic-Based Modules. In: 24th European Conference on Artificial Intelligence (ECAI). pp. 784–791 (2020)
7. Jiménez-Ruiz, E., Cuenca Grau, B.: LogMap: Logic-based and Scalable Ontology Matching. In: Int'l Sem. Web Conf. (ISWC). pp. 273–288 (2011)
8. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Logic-based assessment of the compatibility of UMLS ontology sources. J. Biomed. Sem. 2 (2011)
9. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: Europ. Conf. on Artif. Intell. (ECAI) (2012)
10. Jiménez-Ruiz, E., Meilicke, C., Cuenca Grau, B., Horrocks, I.: Evaluating mapping repair systems with large biomedical ontologies. In: 26th Description Logics Workshop (2013)
11. Solimando, A., Jimenez-Ruiz, E., Guerrini, G.: Minimizing conservativity violations in ontology alignments: Algorithms and evaluation. Knowledge and Information Systems (2016), https://github.com/asolimando/logmap-conservativity/
12. Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: Starspace: Embed all the things! arXiv preprint arXiv:1709.03856 (2017)

OntoConnect: Results for OAEI 2020

Jaydeep Chakraborty[1], Beyza Yaman[2], Luca Virgili[3], Krishanu Konar[4], and Srividya K. Bansal[1]

[1] CIDSE, Arizona State University, Tempe, Arizona
[2] ADAPT Centre, Dublin City University, Dublin, Ireland
[3] Polytechnic University of Marche, Ancona, Italy
[4] Media.net, Mumbai, India

Abstract. The results of OntoConnect, an Ontology alignment system, in the Ontology Alignment Evaluation Initiative (OAEI) 2020 campaign is reported in this paper. OntoConnect is a domain-independent schema alignment system that combines syntactic similarity and structural similarity between classes/concepts to align the classes/concepts from the source and target ontologies. This paper describes the participation of OntoConnect at OAEI 2020 and discusses its methodology and results on the Anatomy dataset.

Keywords: Ontology alignment · Ontology Matching · Unsupervised Learning · Recursive Neural Network.

1 Presentation of the system

OntoConnect [3] is an ontology alignment system that uses an unsupervised machine learning technique that can predict similar source and target ontology classes based on their ontological structure (hierarchy, meta-information, etc.) and syntactic structure without any background domain knowledge or domain expert intervention in contrast to existing learning-based approaches. In the following sections, we present the methodology behind the system and the results of the system participation in the OAEI initiative.

1.1 State, purpose, general statement

Ontology alignment is a process to integrate multiple knowledge bases to eliminate data heterogeneity. There are many ways to address the ontology alignment problem such as string-based approach, language-based approach, semantic approach, extensional approach, etc. Most of the current state-of-the-art ontology alignment systems depend on domain knowledge that makes the alignment process domain-specific, time-consuming, and error-prone to human error. To overcome this challenge, we developed an ontology alignment approach that is

independent of domain knowledge and does not need the domain expert intervention. In this paper, the OntoConnect ontology alignment system is presented which employs an unsupervised learning method using a recursive neural network to align classes between different ontologies.

1.2 Specific techniques used

OntoConnect consists of two main tasks: the first task is unsupervised learning of the OntoConnect model with source ontology classes/concepts. The second task is the prediction of similar source classes/concepts for the corresponding target ontology class/concept using the trained OntoConnect model. Figure 1 represents a workflow of the proposed OntoConnect ontology alignment system.

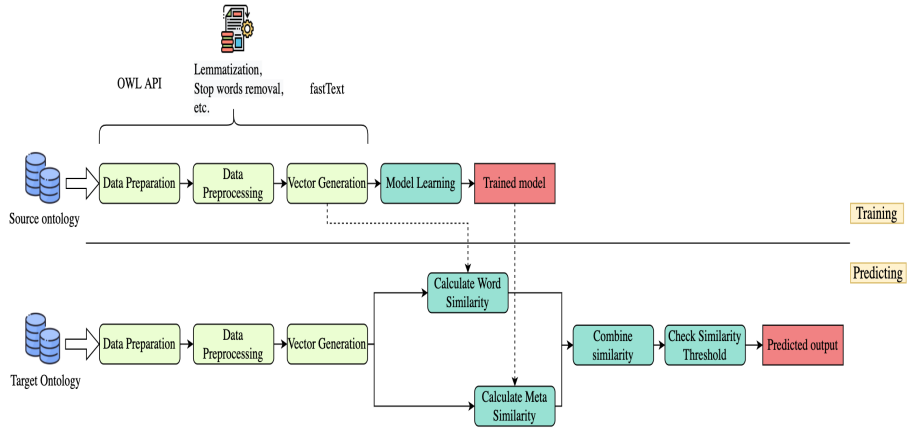


Fig. 1. Overview of OntoConnect Ontology Alignment system

(i) ***Data Preparation:*** In this step, a Java API named OWL API [8] and HermiT Reasoner [9] are used to extract meta information of a class/concept, such as IRI, label, restriction, parent, child, equivalent, and disjoint classes of each class/concept of the source ontology (S) and the target ontology (T).

(ii) ***Data Preprocessing:*** Several data preprocessing techniques are used on both source and target class/concept labels. Special characters and common stop-words in English are removed from the class/concept labels. Apart from stopword, we have used tokenization, lemmatization, conversion of roman letters to numeric, etc.

(iii) ***Vector Generation:*** In this step, a pre-trained embedding model called fastText [2] developed by Facebook's AI Research (FAIR) lab is used on the

source and the target ontology class/concept to generate vectors. It treats each word as composed of character n-grams. So the vector for a word is made of the sum of this character n-grams. It helps to get a meaningful vector even when the dictionary word is not present in the model. The default dimension of the generated vector is 300.

(iv) ***Model Learning:*** Next, the vector generated for each source ontology class/concept is fed to an unsupervised recursive neural network [4]. The recursive neural network is an extension of a recurrent neural network [5]. The input to the recursive neural network is the meta-information of a source ontology class and the output is the source ontology class itself. The intuition behind this learning process is that during prediction if any target class has meta information similar to a source ontology class meta information then the model will be able to predict the same/similar vector to the source ontology class. Figure 2 shows the general architecture of the recursive neural network in OntoConnect. In the figure, $pc_1...pc_m$ denote the parent classes of a class/concept. Similarly, $cc_1...cc_n$, ec_1, dc_1, $rc'_1...rc'_s...rc''_1...rc''_t$ are child classes, equivalent class, disjoint class, and restriction classes of a class/concept. $X(pc_1)$ is the vector representation of pc_1 obtained from pre-trained fastText model. $c(pc_1)$ is the cell state and $h(pc_1)$ is the hidden state of the long short term memory (LSTM) cell [7] for parent meta information. At the output level, the model generates a vector with the same dimension as that of the input vector.

(v) ***Model Prediction:*** The word similarity is calculated by the cosine similarity between the source and target class/concept vectors. Next, the meta-information of the target ontology class is fed to the trained ontology alignment model which predicts a vector similar to one of the source classes. We use the cosine similarity to measure the meta similarity as well. A combined similarity i.e., the average of the word similarity and meta similarity, is used for the final prediction of similar class/concept.

1.3 Adaptations made for the evaluation

OntoConnect consists of two components. The first one is the java component and the second one is the python component. Figure 3 shows a high-level system architecture of OntoConnect. It follows a microservices architecture, consisting of different components that work together. The main motivation behind using microservices was to isolate different tasks and use some of the existing modules within our project. This allowed the use of different programming languages for different purposes based on their applicability. Each microservice was dockerized, making it modular, portable, as well as isolating the environments so as to run on any operating system.

We have tried to test the OntoConnect system on Semantic Evaluation At Large Scale (SEALS) [11] platform, however, were not able to run the system as SEALS only provides a wrapper for java-specific tools only. Other frameworks

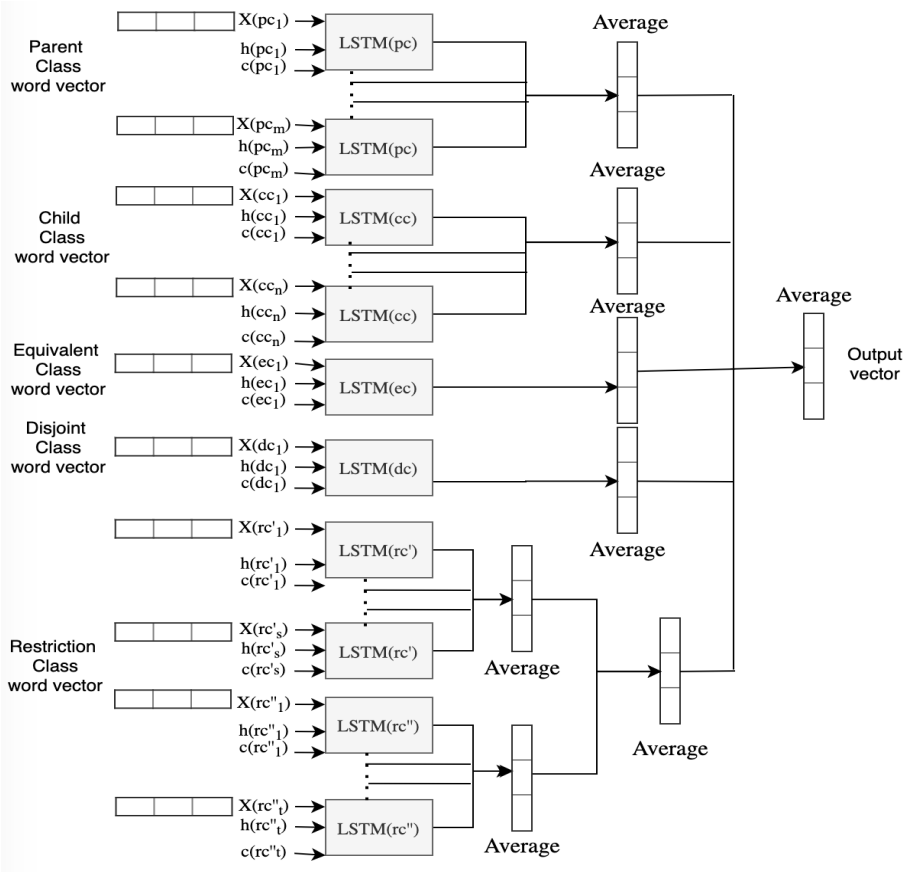


Fig. 2. Recursive Neural Network (dynamic array tree-LSTM model) of Ontology Alignment System

such as MELT [6] was also tried for the evaluation of the OntoConnect System, however, MELT provides an evaluation wrapper for either java-only tools or python-only tools. It does not support tools that have both java and python components in one. OntoConnect system uses both the java and python components. Hobbit platform [10] permits dockerized tool which is independent of the type of the programming language of the tool. For this reason, the dockerized approach is used to build the OntoConnect System and we could successfully test and evaluate it on the Hobbit platform.

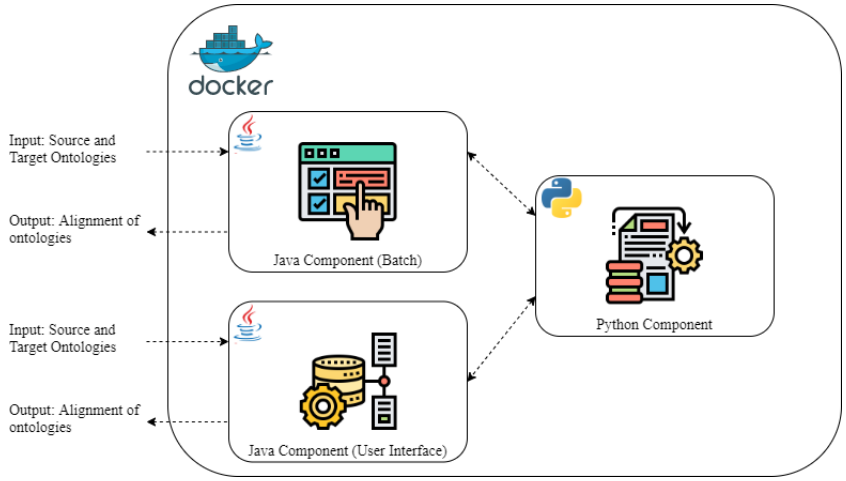


Fig. 3. OntoConnect system architecture

1.4 Link to the system and parameters file

The OntoConnect code is available on GitHub: `https://github.com/dbpedia/linking`

1.5 Link to the set of provided alignments

The OntoConnect result is published on `http://oaei.ontologymatching.org/2020/results/anatomy/index.html` . The result is also available on GitHub: `https://github.com/dbpedia/linking/wiki/Result`

2 Results

We have tested OntoConnect on the Anatomy [1] data set published by OAEI with different parameters such as input vector dimension and similarity threshold. Three different files are provided in the OAEI System: source ontology,

target ontology, and result or alignment file. Standard evaluation metrics, i.e., precision, recall, and F-measure are used. The OntoConnect system yields satisfactory results with a precision of 99.6%, recall of 66.5%, and F-measure of 79.7% for a similarity threshold of 0.99 with the 100-dimension input vector. Table 1 gives a summary of the result of OntoConnect on the Anatomy data set.

Table 1. OntoConnect performance in the Anatomy track

Matcher	Runtime	Precision	Recall	Recall++	F-Measure
OntoConnect	248	0.996	0.665	0.136	0.797

3 General comments

The main goal of the OntoConnect is to address questions such as, (i) can ontology alignment be done independently of domain information? (ii) Can ontology alignment be achieved by using only the meta-information and structural information of ontologies? (iii) Can ontology alignment be achieved using unsupervised machine learning instead of the traditional rule-based approaches? The OntoConnect tool is able to address all the above questions and moreover, it performs well compared to some of the state-of-the-art systems in OAEI 2020. The main strength of the tool is that a domain-independent approach is performed by achieving the mentioned goals.

Besides the strengths of the tool, there is a number of potential improvements to be realized for OntoConnect. The main weakness of the OntoConnect tool is the complex architecture of the system, as it has two different components of different languages i.e. java and python. It was difficult to incorporate any OAEI evaluation wrapper because of the complex architecture of the tool. We have used Docker to execute the system on the HOBBIT platform but there is still room for improving the system architecture so that the tool can be easily executed. The second problem is the size of the project. We have used the pre-trained model fastText in the system and the default dimension of the fastText output vector is 300. The high dimension of the vector causes an increase in the size of the tool. In future work, we would like to explore different procedures such as autoencoder approach to reduce the dimension to minimize the size of the tool.

4 Conclusion

In this study, OntoConnect tool is presented with a generic and domain-independent approach to align multiple ontologies that eliminate cumbersome and error-prone manual work. A non-linear neural network is used for feature extraction from the source ontology and is independent of the domain knowledge. Participating in

this campaign for the first time allowed us to see how the OntoConnect system was performing compared to the other tools. It was seen that our tool had a high precision among the tools without any domain knowledge and without depending on any vocabularies. But both recall and F1 have room to improve. Even though OntoConnect has a reasonable runtime, we would like to decrease the execution time for better performance. We have seen that our tool is comparable to the current state-of-the-art domain-specific approaches and we would like to participate in other tracks next year to see the results in different domains.

Acknowledgement The authors gratefully acknowledge the Google Summer Code program and DBpedia organization for guidance and support. We also thank the Google Cloud Platform (GCP) research credits program for providing an environment to run the experiments using their Cloud Computing services.

Beyza Yaman has been supported by the European Union's Horizon 2020 research and innovation programme under Marie Sklodowska-Curie grant agreement No. 801522, by Science Foundation Ireland and co-funded by the European Regional Development Fund through the ADAPT Centre for Digital Content Technology [grant number 13/RC/2106] and Ordnance Survey Ireland.

References

1. http://oaei.ontologymatching.org/2020/anatomy/index.html
2. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)
3. Chakraborty, J., Bansal, S., Yaman, B., Virgili, L., Konar, K.: Ontoconnect: Unsupervised ontology alignment with recursive neural network. In: Proceedings of the 36th ACM/SIGAPP Symposium on Applied Computing, SAC 2021, Gwangjiu, South Korea,March 22-26, 2021 (In Press)
4. Chinea, A.: Understanding the principles of recursive neural networks: a generative approach to tackle model complexity. In: International Conference on Artificial Neural Networks. pp. 952–963. Springer (2009)
5. Goller, C., Kuchler, A.: Learning task-dependent distributed representations by backpropagation through structure. In: Proceedings of International Conference on Neural Networks (ICNN'96). vol. 1, pp. 347–352. IEEE (1996)
6. Hertling, S., Portisch, J., Paulheim, H.: Melt-matching evaluation toolkit. In: International Conference on Semantic Systems. pp. 231–245. Springer, Cham (2019)
7. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
8. Horridge, M., Bechhofer, S.: The owl api: A java api for owl ontologies. Semantic web **2**(1), 11–21 (2011)
9. Motik, B., Shearer, R., Horrocks, I.: Optimized reasoning in description logics using hypertableaux. In: International Conference on Automated Deduction. pp. 67–83. Springer (2007)
10. Röder, M., Kuchelev, D., Ngonga Ngomo, A.C.: Hobbit: A platform for benchmarking big linked data. Data Science (Preprint), 1–21 (2019)
11. Wrigley, S.N., García-Castro, R., Nixon, L.: Semantic evaluation at large scale (seals). In: Proceedings of the 21st International Conference on World Wide Web. pp. 299–302 (2012)

RE-miner for data linking results for OAEI 2020*

Armita Khajeh Nassiri[1 [0000−0002−5734−0351]], Nathalie Pernelle[1,2 [0000−0003−1487−393X]], Fatiha Saïs[1 [0000−0002−6995−2785]], and Gianluca Quercini[1 [0000−0001−9195−1618]]

[1] LRI, CNRS 8623, Paris Saclay University, Orsay F-91405, France
[2] LIPN, CNRS (UMR 7030), University Sorbonne Paris Nord, France
firstname.lastname@lri.fr

Abstract. This paper presents the RE-miner results for data linking in the ontology alignment contest OAEI 2020, Spimbench track. RE-miner discovers all minimal and diverse referring expressions of all instances of a given source knowledge graph. In a second step, it exploits these referring expressions to find the possible links to a target knowledge graph. This is the first participation of RE-miner in the OAEI campaign and produces the best result in terms of F-measure on the Spimbench dataset.

1 Presentation of the system

As the Web of Data continues to grow, more and more knowledge graphs (KGs) that cover a wide range of topics are emerging in the Linked Open Data (LOD) Cloud. As knowledge graphs are usually built independently from one another, inevitably, the same Internationalized Resource Identifier (IRI) is not necessarily reused for a given individual. Thus, it is essential to have systems capable of data linking, i.e., to produce a set of mapping between the individuals of two knowledge graphs representing the same real-world object. RE-miner for data linking is one such system that, given a subset of class and property mappings between the source and target knowledge graphs, identifies possible sameAs links between the instances of the two KGs.

1.1 State, purpose, general statement

RE-miner for data linking consists of 2 main steps. The algorithm has been thoroughly presented in [4]. Here, we will miss out on the details and present the major steps taken in this campaign. First, discovering referring expressions for all instances of the source knowledge graph. A referring expression (RE) is a description that identifies an instance unambiguously in a class of a knowledge graph—instantiating the keys of a class yields numerous REs itself. However, many more referring expressions can potentially be found. To reduce the search space, RE-miner focuses on non-key properties. Both keys and maximal non-keys are obtained using SAKey [5]. Second, all the REs

discovered on a class of source knowledge graph are taken into account to link to instances of a target KG. The idea behind using REs for linking is that if an instance x in the target knowledge graph satisfies a description that uniquely identifies the instance u in the source knowledge graph, it is probable that the two instances are the same. Using different referring expressions, an instance u might be linked to different target KG instances. A voting strategy is employed to choose the most confident link whenever possible.

1.2 Specific techniques used

This system focuses on the instance matching problem between the instances of a given class of the source dataset, on which the REs have been discovered, and a target dataset having a non-empty set of mapped properties to the source. In other words, this approach assumes the schemas to have previously been aligned.

Create the source dataset. We first create the dataset on the source KG for the given class C, for which we aim to find the alignments. The dataset is created by keeping all instances that are of type C, and all sub-classes of C if the graph's schema is not saturated. For instance, in the Spimbench track, the instances of *Creative Works* class are to be linked. The dataset, contains all instanced belonging to this class and its 3 sub-classes namely *NewsItem*, *BlogPost*, and *Programme*.

Referring Expressions. We discover all minimal and diverse referring expressions of depth 1 on the source knowledge graph [4]. These REs do not contain the existential quantifier and are conjunctions of atoms (e.g., $album(x) \wedge createdBy(x, Beatles) \wedge releasedOn(x, \text{“}1966-05-2\text{”})$ holds as a referring expression when x is instantiated with Yellow Submarine). We enrich this set, with the set of referring expressions that are obtained through instantiating each set of key properties for class C obtained using SAKey. Being a referring expression, each of these descriptions, holds only for one instance in the class C of the source KG.

Linking and Voting Strategy. These REs are then used to find possible links in the target dataset. For finding the possible candidate links, mapped properties and strict equality are used between the atoms of a RE and triples of the target knowledge graph. Moreover, first consider an instance u of type C in the source dataset and imagine that k different referring expressions $\{RE_1(u), ..., RE_k(u)\}$ have been associated to it. Each of these REs can be linked to zero, one, or more instances of the target, using the bottom-up approach explained in [4]. We consider the properties mapped if they are strictly equal in source and target.
The confidence of each RE is inverse proportional to the number of links it suggests. However, if the unique name assumption (UNA) is fulfilled, only one sameAs link can be found between u and an instance x belonging to the target KG. Thus we propose a voting strategy that assigns a weight to each distinct link. The weight is the sum of the confidence degree of the REs proposing that link. Moreover, the weights are normalized such that they have a value between 0 and 1. Finally, the instance x in the target knowledge graph being linked to u with the highest weight is selected. For the Spimbench

dataset, we have set a very strict criterion. We only match two instances if and only if the link with the highest weight has a weight equal to one. This way, we imply that we only link two instances if we are really sure about it.

MELT. Matching EvaLuation Toolkit (MELT) is a framework optimized for OAEI campaigns, facilitating submissions to the SEALS and HOBBIT evaluation platforms [2]. The Spimbench track, on which we evaluate our performance, is available on the HOBBIT, Holistic Benchmarking of Big Linked Data, platform[1]. We used MELT to wrap it as a HOBBIT package, and as our implementation is in Python, we used MELT's External Matching. Thankfully, MELT has eased the submission process; however, we assume that it causes some run-time overhead.

2 Results

2.1 Spimbench track

Spimbench is an instance matching track and the only track we have done evaluations on, in this first year of participation. It consists of two datasets of different sizes: the SANDBOX dataset with about 380 instances and 10000 triplets, and the MAINBOX dataset with about 1800 instances and 50000 triplets. We have compared our results with AML [1], Lily [7], FTRL_IM [6], and LogMap [3] in Table 2.1. All these systems had participated in the past year(s) of the competition.

Table 1. Comparison of Performance in Spimbench track. The time performance is reported in ms.

		Precision	Recall	F-measure	Time
SANDBOX	AML	0.8348	0.8963	0.8645	6446
	Lily	0.9835	**1.0**	0.9917	2050
	FTRL-IM	0.8542	**1.0**	0.9214	**1525**
	LogMap	0.9382	0.7625	0.8413	7483
	RE-miner	**1.0**	0.9966	**0.9983**	7284
MAINBOX	AML	0.8385	0.8835	0.8604	38772
	Lily	0.9908	**1.0**	0.9953	3899
	FTRL-IM	0.8558	0.9980	0.9214	**2247**
	LogMap	0.8801	0.7094	0.7856	26782
	RE-miner	**0.9986**	0.9966	**0.9976**	33966

The same strategy explained in Section 1.1 is used on both datasets for RE-miner. In total, for the Sandbox dataset, 6920 REs are created. Whereas for the Mainbox dataset, there are a total of 39892 REs among which 14085 are from key instantiation. We can observe that we outperform the other systems in terms of Precision, and F-measure on both datasets, showing a slight better performance than Lily. However, we come

[1] `http://project-hobbit.eu/`

short when comparing the time-performance. This is mainly due to the fact that our system must first compute the keys and non-keys of a given class using a Java-based application, and then find the REs. Indeed more optimization can be done to decrease the run-time.

3 General Comments

RE-miner for data linking has shown satisfactory results in the Spimbench instance matching track. Although the source and target KGs shared almost the same ontology, there were still some properties that would not be mapped together using strict similarity. However, this did not hamper the performance of our system. This is because of the fact that RE-miner usually discovers not just one but many more REs for each instance. This will allow the system to choose the target instance most of the REs pointing to agree on. Moreover, for this dataset, we have been fastidious, only outputting links we really deem correct. As future work, we aim to do modifications, allowing us to participate in more tracks for the next years and focus more on enhancing our system's run-time.

4 Conclusion

In this paper, we briefly presented the main components of our instance matching system RE-miner for data linking. The evaluation of results on the Spimbench track was presented, and we showed a better Precision and F-measure than other systems taking part in the campaign this year. However, in terms of run-time, more improvement and optimization are to be done.

References

1. Faria, D., Pesquita, C., Tervo, T., Couto, F.M., Cruz, I.F.: AML and AMLC results for OAEI 2019. In: Shvaiko, P., Euzenat, J., Jiménez-Ruiz, E., Hassanzadeh, O., Trojahn, C. (eds.) Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019. CEUR Workshop Proceedings, vol. 2536, pp. 101–106. CEUR-WS.org (2019), http://ceur-ws.org/Vol-2536/oaei19_paper3.pdf
2. Hertling, S., Portisch, J., Paulheim, H.: MELT - matching evaluation toolkit. In: Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9-12, 2019, Proceedings. pp. 231–245 (2019)
3. Jiménez-Ruiz, E.: Logmap family participation in the OAEI 2019. In: Shvaiko, P., Euzenat, J., Jiménez-Ruiz, E., Hassanzadeh, O., Trojahn, C. (eds.) Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019. CEUR Workshop Proceedings, vol. 2536, pp. 160–163. CEUR-WS.org (2019), http://ceur-ws.org/Vol-2536/oaei19_paper11.pdf
4. Khajeh Nassiri, A., Pernelle, N., Saïs, F., Quercini, G.: Generating referring expressions from rdf knowledge graphs for data linking. In: Pan, J.Z., Tamma, V., d'Amato, C., Janowicz, K., Fu, B., Polleres, A., Seneviratne, O., Kagal, L. (eds.) The Semantic Web – ISWC 2020. pp. 311–329. Springer International Publishing, Cham (2020)

5. Symeonidou, D., Armant, V., Pernelle, N., Saïs, F.: Sakey: Scalable almost key discovery in rdf data. In: International Semantic Web Conference. pp. 33–49. Springer (2014)
6. Wang, X., Jiang, Y., Luo, Y., Fan, H., Jiang, H., Zhu, H., Liu, Q.: FTRLIM results for OAEI 2019. In: Shvaiko, P., Euzenat, J., Jiménez-Ruiz, E., Hassanzadeh, O., Trojahn, C. (eds.) Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019. CEUR Workshop Proceedings, vol. 2536, pp. 146–152. CEUR-WS.org (2019), http://ceur-ws.org/Vol-2536/oaei19_paper9.pdf
7. Wu, J., Pan, Z., Zhang, C., Wang, P.: Lily results for OAEI 2019. In: Shvaiko, P., Euzenat, J., Jiménez-Ruiz, E., Hassanzadeh, O., Trojahn, C. (eds.) Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019. CEUR Workshop Proceedings, vol. 2536, pp. 153–159. CEUR-WS.org (2019), http://ceur-ws.org/Vol-2536/oaei19_paper10.pdf

VeeAlign: A Supervised Deep Learning Approach to Ontology Alignment*

Vivek Iyer[1], Arvind Agarwal[2], and Harshit Kumar[2]

[1] IIIT Hyderabad, India
vivek.iyer@research.iiit.ac.in
[2] IBM Research, India
{arvagarw,harshitk}@in.ibm.com

Abstract. [3]While deep learning approaches have shown promising results in Natural Language Processing and Computer Vision domains, they have not yet been able to achieve impressive results in Ontology Alignment, and have typically performed worse than rule-based approaches. Some of the major reasons for this are: a) poor modelling of context, b) overfitting of standard DL models, and c) dataset sparsity, caused by class imbalance of positive alignment pairs wrt negative pairs. To mitigate these limitations, we propose a dual-attention based approach that uses a multi-faceted context representation to compute contextualized representations of concepts, which is then used to discover semantically equivalent concepts.

Keywords: Ontology Alignment · Deep Learning.

1 Presentation of the System

The task of ontology alignment aims to determine correspondences between semantically related concepts - classes and properties- across two ontologies. OAEI [3] has been conducting ontology alignment challenges since 2004 where multiple datasets belonging to different domains are released along with a public evaluation platform to evaluate different matching systems. Matching systems that have been proposed can broadly be classified into two types: rule based systems [4, 7] and statistical based systems [5, 8, 9, 11]

Rule based system uses handcrafted rules with manually assigned weights, coupled with various string similarity algorithms to discover concept alignments, often utilizing domain-specific knowledge as well. This kind of approach, while easy to implement, has some obvious limitations: a) Using string similarity algorithms with minimal focus on context does not address either semantic or contextual relatedness. b) For every pair of ontologies, a new set of rules and weights may need to be defined, which is often a laborious and time consuming process, thus adversely affecting scalability.

* This work was done while Vivek Iyer was an intern at IBM Research, India

Deep Learning models proposed so far [11] have used external information to enrich entities, such as synonyms from the ontology, definitions from Wikipedia and context from background knowledge sources. Along similar lines, DeepAlign[9] uses external information, such as synonyms and antonyms extracted from WordNet and PPDB, for refining word vectors, which are then used for alignment. Despite the uniqueness of the proposed approaches, Deep Learning models have typically been unable to thrive in the task of Ontology Alignment, and have performed worse than rule-based models. Some of the major reasons for this are: a) lack of focus on ontological structure, and thus poor modelling of context, b) overfitting of standard DL models, and c) dataset sparsity, caused by class imbalance of positive alignment pairs wrt negative pairs. This occurs because the number of positive alignments is typically several orders smaller than the number of negative alignments.

1.1 State, Purpose and General Statement

In an effort to mitigate the above-mentioned challenges, we propose VeeAlign[6], an ontology alignment system that aligns classes and properties based on not just semantic, but also structural similarity which is driven by its context. Our method thus incorporates a novel way of modelling context, where it is split into multiple facets based on the type of neighborhood. Based on their relative importance, some of these facets include only neighbouring one-hop nodes, while others also include the paths from the root to these nodes. To address this challenge, we use a novel dual attention mechanism that comprises of path level attention followed by node level attention. The former helps find the most important path among all the available paths, while the latter finds the node with the greatest influence on the alignment of the central concept.

1.2 Specific Techniques Used

VeeAlign[6] is a supervised Deep Learning based ontology alignment system, that computes a contextualized representation of concepts as a function of not just its label but also its multi-faceted neighbouring concepts surrounding it. In other words, the context is divided into multiple facets based on the relationship between the concept and its neighbours, and then a contextual vector is computed using a dual attention mechanism. This contextual vector is later concatenated with semantic distribution of the concept label, thus computing a contextualised concept representation which is later used to discover alignments. Figure 1 shows the architecture diagram of the proposed system. Thus, the key significance of the proposed approach is that VeeAlign exploits not just the semantic aspect, like previous approaches, but also the syntactical structure of ontologies and uses that alone to achieve state-of-the-art results, without any requirement for background knowledge.

Context Representation In VeeAlign, in the case of concepts, its neighboring concepts form its context. Each neighboring concept has a role and exerts a

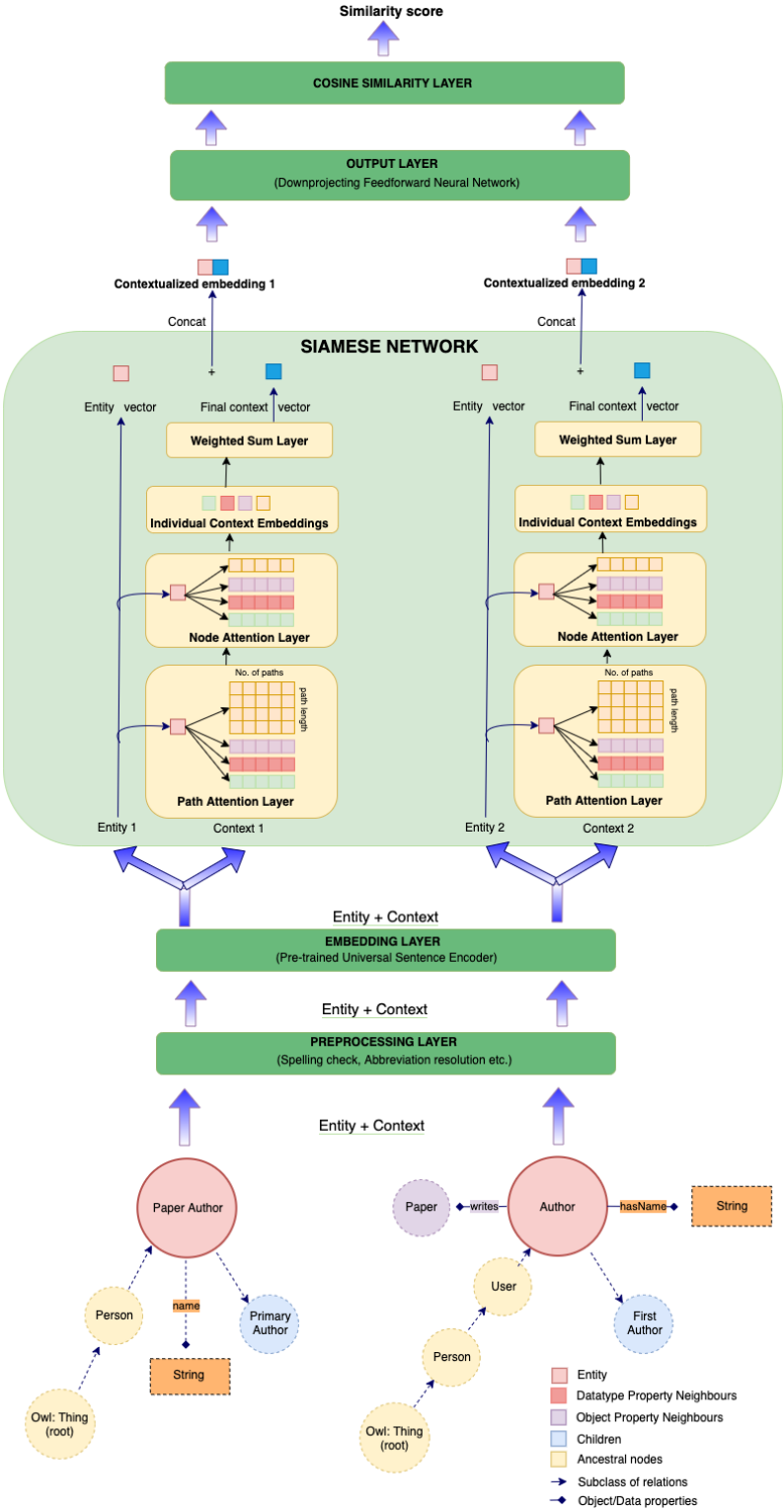


Fig. 1: VeeAlign Architecture

separate influence on the concept alignment, therefore we categorize neighboring concepts into four facets: ancestor nodes, child nodes, nodes connected through a datatype property and nodes connected through an object property. Here, we define parent and child nodes as those connected to the central node through SUBCLASS-OF relations. Sifting through several ontologies and their reference alignments, we observed that two concepts align not just based on their one-hop neighbours, but also on the basis of similarity of "ancestral nodes". Ancestral nodes of a concept are defined as the nodes that lie on the path all the way from the root to the parent node of the concept, and this path is referred to as a "lineage path". Thus, given a concept, apart from its one-hop neighbours we also enumerate all its lineage paths, consider them as part of context and use them while computing alignments. Since only ancestors will have such "lineage paths" and not one-hop neighbours (children, datatype property neighbours and object property neighbours), we consider these one-hop neighbours as paths of length 1 in order to maintain consistent terminology. The context of properties is modelled separately, by considering their domain and range as context.

Dual Attention Attention [1, 10] in deep learning can be broadly interpreted as a vector of weights denoting relative importance. For the task at hand, attention weighs neighboring concepts in proportion with their influence on the central concept's alignment. The attention process used to compute weights consists of a dual attention mechanism: first at the path level, referred to as Path-level attention, and the next at the node level, referred to as Node-level attention. The goal is to assign higher weights to the more influential paths using Path-level attention. And, within the most influential path, higher weights are assigned to nodes that are the most influential.

Thus, path level attention aims to find the most important paths for each contextual facet. This involves computing the attention weight of each node in each path with respect to the main concept, and then adding and normalizing these weights. When done for each lineage path, it yields the relative importance of that path. Given the relative importance of each path, a max-pool layer is then applied over each of these path weights to yield the most important path. After path-level attention, the next step is node-level attention. This is achieved by computing the attention weights of each node in the most important path. These weights are then used to take a weighted linear combination of the node embeddings available in the path embedding.

Model Training As shown in Figure 1, the training process involves computing the representations of all four types of context i.e., ancestral context, child context, data property neighbour context and object property neighbour context. The context representation computation involves applying the path-level attention followed by the node-level attention. The child, data properties and object properties context have paths of only length one, therefore there is only path level attention, not the node level attention. These four context representations are combined through a weighted linear combinations to get the final

context representation. This context vector is then concatenated with the semantic representation of the central concept, and the combined representation is input to a linear layer for dimensionality reduction in a lower dimension space. Since a candidate alignment pair consists of elements (concepts or properties) from both source and target ontologies, we perform the above computations for both source and target elements to get the contextualized representations by passing both through a Siamese Network [2], and then compute the confidence score of the alignment by taking a cosine similarity between the two contextualized representations. For the alignment of properties, we consider its context as its domain and range, and obtain the confidence score as a weighted sum of the similarities between the respective distributional embeddings of domains, ranges and property labels respectively. Finally, an element pair is considered as a positive alignment when the similarity score is more than an experimentally determined threshold, and discarded otherwise. We use mean squared loss computed between the predicted and ground truth labels for training our model.

1.3 Datasets and Experimental Setting

Our system requires training data in the form of positive and negative alignment pairs. Although many tracks in OAEI have reference alignments that contain positive example pairs, as per the rules, one could not use them for training. So, we resorted to using pseudo-training data, i.e the output of the highest-performing system, AML, as an approximation of the reference alignments. However, we found that AML was only able to identify concepts whose names had some sort of string similarity, but not concepts that had different names but similar contexts. Our approach of modelling structural context and then attending on it thrives on identifying not just the former using semantic similarity, but also the latter using structural similarity. Since this was missing in AML's output, there were a large number of False Negatives (FNs). In addition, there were a few False Positives (FPs) with similar names but different contexts. Thus instead of using AML's output directly, we decided to manually correct AML's output as a preliminary stage. In order to discover FPs, we simply validated AML's output and discovered 12 FPs. To discover FNs, we took a cartesian product of all concept pairs, sorted them based on context similarity and annotated the top 1000 pairs, discovering a total of 35 FNs. Our annotations process included three annotators where the final decision was taken using majority voting algorithm.

As part of our submission, we targeted two tracks i.e. Conference and Multifarm. We performed the above exercise for all ontology pairs in the Conference track. For Multifarm, due to lack of time, we could not train a separate model or integrate a translator, but instead used the pre-trained Conference model, since both tracks share similar ontological structure and are comprised of the same ontologies, albeit in different languages. In place of a translator, we merely

Table 1: Conference track results of OAEI 2020

	ra1-M1	ra1-M2	ra1-M3	ra2-M1	ra2-M2	ra2-M3	rar2-M1	rar2-M2	rar2-M3
VeeAlign	**0.78**	0.34	0.73	**0.74**	0.34	0.69	**0.74**	0.34	**0.7**
AML	0.76	**0.58**	**0.74**	0.71	**0.58**	**0.7**	0.71	**0.56**	0.69
LogMap	0.73	0.39	0.69	0.67	0.39	0.63	0.69	0.4	0.66
Wiktionary	0.69	0.26	0.61	0.64	0.26	0.57	0.65	0.27	0.58
ATBox	0.69	0.23	0.6	0.64	0.26	0.56	0.65	0.26	0.57
ALOD2Vec	0.68	0.25	0.59	0.62	0.25	0.55	0.64	0.26	0.56
LogMapLt	0.66	0.23	0.59	0.6	0.23	0.54	0.62	0.23	0.56
ALIN	0.67	-	0.6	0.61	-	0.55	0.63	-	0.56
edna	0.67	0.14	0.59	0.61	0.14	0.54	0.63	0.14	0.56
StringEquiv	0.64	0.03	0.56	0.58	0.03	0.52	0.61	0.03	0.53
Lily	0.62	-	0.55	0.57	-	0.5	0.57	-	0.51
DESKMatcher	0.25	0.02	0.18	0.23	0.02	0.16	0.23	0.02	0.16

substituted the initial Universal Sentence Encoder[4] model with its multilingual variant[5].

Our Deep Learning model requires several parameters and hyperparameters, which we optimized through a grid-search algorithm. The model was converged using MSE loss and Adam optimizer with a learning rate of 0.001, after training for 50 epochs with a batch size of 32. The maximum number of paths considered for each node is set to 21, and the maximum length of the path is taken as 8. All randomizations including the ones in PyTorch and Numpy are done by setting 0 as the manual seed.

1.4 Link to the System and Parameters File

The entire codebase of our system is available on GitHub[6] and so is the configuration parameters file[7].

2 Results and Discussions

2.1 Conference

Table 1 shows the results of the conference track of OAEI 2020. There are in total 9 matching tasks. There are three different reference alignments: ra1 (original), ra2 (consistency violation-free) and rar2 (consistency & conservativity violation-free), each of which contain class matching, property matching and hybrid (both class and property) matching tasks respectively. VeeAlign achieves state-of-the-art results in all the entity matching tasks, 3rd position in all the property matching tasks and top 2 positions in all the hybrid matching tasks, sometimes losing out to AML by a narrow margin of 0.01 points in F1-score. In rar2, which

[4] https://tfhub.dev/google/universal-sentence-encoder-large/5
[5] https://tfhub.dev/google/universal-sentence-encoder-multilingual/3
[6] https://github.com/Remorax/VeeAlign
[7] https://github.com/Remorax/VeeAlign/blob/master/src/config.ini

Table 2: Multifarm track results of OAEI 2020

System	Different ontologies	Same ontologies
AML	**0.47**	0.28
LogMap	0.37	**0.41**
VeeAlign	0.15	0.14
Wiktionary	0.32	0.12
LogMapLt	0.04	0.01

has been considered to be the main set of reference alignments this year (since it is both consistency and conservativity violation-free), VeeAlign tops the table.

2.2 Multifarm

Table 2 shows the multifarm track results. As expected, VeeAlign is unable to compete with state-of-the-art systems in this track, but still manages to produce decent results when one factors in the lack of a separately trained model and more importantly, a translator.

3 General comments

3.1 Comments on results

VeeAlign has certainly produced some very encouraging results. In the conference track, it achieves state-of-the-art results in entity matching, which indicates that our two-step attention model that adopts a multifaceted approach to context representation, is able to outperform systems that use handcrafted rules and manually assigned parameter values. VeeAlign's performance dips when it comes to property matching, indicating that our approach of modelling property similarity as a weighted sum of property, domain and range similarity is not good enough and needs more work. Nevertheless, our current approach ensured we achieve state-of-the-art results in hybrid-matching tasks, a sizeable feat in just the debut run.

In the multifarm track, unsurprisingly VeeAlign found itself being unable to compete with other SOTA systems, that, unlike VeeAlign, used an integrated translator. However despite this and the lack of a separately trained model, it was still able to use ontological structure to produce moderately decent results with low recall but high precision.

3.2 Comments on OAEI measures

While current OAEI evaluation measures are definitely thorough and multifaceted, a possible improvement would be to incorporate K-fold sliding window evaluation. Here, (K-1) folds of the reference alignments are provided as input (if the system needs it for training) and the last fold is used for testing. This is done

for every possible fold, and an average of the performance on each fold is taken to yield the final performance. This data split into K-folds can occur either at the ontology-pair level (possible in tracks where there are a large number of ontology pairs in the reference alignments, such as conference or multifarm tracks) or at the "concept-pair" level (possible in tracks where there are fewer number of ontology pairs, but the ontologies are larger in size). In case of the former, (K-1) folds of ontologies are provided for training and 1 fold for testing. In the conference track, for instance, which has 21 ontology pairs , if we take K=7, 6 folds (i.e 18 ontology pairs) are provided for training and the last fold (i.e 3 ontology pairs) are tested on. This is repeated for every 3 pairs of ontologies, and an average is taken.

In case of the latter, a "concept-pair" split could be achieved by taking a cartesian product of the concepts in both ontologies, splitting it into K folds, taking K-1 for training and in each test fold, the concept pairs which are part of the alignment output by a system can be marked as True, and the rest as False. The same can be done for the ground truth alignments, and the corresponding True Positives (TPs), False Positives (FPs) and True Negatives (TNs) can be calculated to yield precision, recall and F1-scores. When evaluating by "concept-pair" split, the input for non-supervised systems remains the same, and is still the entirety of the ontology. For supervised systems, it would be the (K-1) folds of concept pairs, with pairs present in reference alignments being marked as True and the rest as False. Such a file could be dynamically created and input to the system. However, while evaluating the alignments generated by each system, the procedure remains the same, i.e calculating TPs, FPs and TNs on the last fold.

Both of these proposed methods of evaluation are in compliance with OAEI rules, and the training and testing data are clearly separated (removing any chance of over-fitting) and an average is taken to remove any chance of bias. Moreover, K-fold sliding window evaluation is widely accepted in the AI community as a fair mode of evaluation. Going ahead, these measures would give supervised systems a fair chance to compete against systems that use handcrafted rules, would encourage more DL-based systems in the future and thus allow Deep Learning to thrive in Ontology Alignment as well.

4 Conclusion & Future Work

As part of OAEI 2020, we introduced VeeAlign, an Ontology Alignment that uses a supervised Deep Learning approach to discover alignments. In particular, it uses a two-step model of attention combined with multi-facted context representation to produce contextualized representations of concepts, which aids alignment based on semantic and structural properties of an ontology. VeeAlign achieves state-of-the-art results in Conference track beating AML, LogMap and other mature systems in just its debut run, which is certainly encouraging and indicative of the validity and effectiveness of our approach. In multifarm track, VeeAlign produces decent results using solely ontological structure. The results have high precision but low recall due to lack of incorporation of a translator,

which we plan to fix in future editions of OAEI. In addition, we plan on improving property matching to further improve our performance on the Conference track. Lastly and mostx importantly, we plan on targeting the biomedical tracks (such as Anatomy, LargeBio and Biodiversity) and adapting VeeAlign to use biomedical background knowledge, if available.

References

1. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. ICLR (2015)
2. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a" siamese" time delay neural network. In: Advances in neural information processing systems. pp. 737–744 (1994)
3. Euzenat, J., Meilicke, C., Stuckenschmidt, H., Shvaiko, P., Trojahn, C.: Ontology alignment evaluation initiative: six years of experience. In: Journal on data semantics XV, pp. 158–192. Springer (2011)
4. Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F., Couto, F.M.: The agreementmakerlight ontology matching system. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". pp. 527–541. Springer (2013)
5. Huang, J., Dang, J., Vidal, J.M., Huhns, M.N.: Ontology matching using an artificial neural network to learn weights. In: IJCAI workshop on semantic Web for collaborative knowledge acquisition. vol. 106 (2007)
6. Iyer, V., Agarwal, A., Kumar, H.: Multifaceted context representation using dual attention for ontology alignment. arXiv preprint arXiv:2010.11721 (2020)
7. Jiang, S., Lowd, D., Kafle, S., Dou, D.: Ontology matching with knowledge rules. In: Transactions on Large-Scale Data-and Knowledge-Centered Systems XXVIII, pp. 75–95. Springer (2016)
8. Jiménez-Ruiz, E., Agibetov, A., Chen, J., Samwald, M., Cross, V.: Dividing the ontology alignment task with semantic embeddings and logic-based modules. arXiv preprint arXiv:2003.05370 (2020)
9. Kolyvakis, P., Kalousis, A., Kiritsis, D.: Deepalignment: Unsupervised ontology matching with refined word vectors. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 787–798 (2018)
10. Paulus, R., Xiong, C., Socher, R.: A deep reinforced model for abstractive summarization. arXiv preprint arXiv:1705.04304 (2017)
11. Wang, L., Bhagavatula, C., Neumann, M., Lo, K., Wilhelm, C., Ammar, W.: Ontology alignment in the biomedical domain using entity definitions and context. In: Proceedings of the BioNLP 2018 workshop. pp. 47–55 (2018)

Wiktionary Matcher Results for OAEI 2020

Jan Portisch[1,2][0000−0001−5420−0663] and Heiko Paulheim[1][0000−0003−4386−8195]

[1] Data and Web Science Group, University of Mannheim, Germany
{jan, heiko}@informatik.uni-mannheim.de
[2] SAP SE Product Engineering Financial Services, Walldorf, Germany
jan.portisch@sap.com

Abstract. This paper presents the results of the *Wiktionary Matcher* in the *Ontology Alignment Evaluation Initiative* (OAEI) 2020. *Wiktionary Matcher* is an ontology matching tool that exploits *Wiktionary* as external background knowledge source. Wiktionary is a large lexical knowledge resource that is collaboratively built online. Multiple current language versions of Wiktionary are merged and used for monolingual ontology matching by exploiting synonymy relations and for multilingual matching by exploiting the translations given in the resource. This is the second OAEI participation of the matching system. *Wiktionary Matcher* has been improved and is the best performing system on the knowledge graph track this year.[3]

Keywords: Ontology Matching · Ontology Alignment · External Resources · Background Knowledge · Wiktionary

1 Presentation of the System

1.1 State, Purpose, General Statement

The *Wiktionary Matcher* is an element-level, label-based matcher which uses an online lexical resource, namely *Wiktionary*. The latter is "[a] collaborative project run by the Wikimedia Foundation to produce a free and complete dictionary in every language"[4]. The dictionary is organized similarly to Wikipedia: Everybody can contribute to the project and the content is reviewed in a community process. Compared to WordNet [2], Wiktionary is significantly larger and also available in other languages than English. This matcher uses *DBnary* [13], an RDF version of Wiktionary that is publicly available[5]. The DBnary dataset makes use of an extended *LEMON* model [7] to describe the data. For this

[4] see https://web.archive.org/web/20190806080601/https://en.wiktionary.org/wiki/Wiktionary

[5] see http://kaiko.getalp.org/about-dbnary/download/

matcher, recent DBnary datasets for 8 Wiktionary languages[6] have been downloaded and merged into one RDF graph. Triples not required for the matching algorithm, such as glosses, were removed in order to increase the performance of the matcher and to lower its memory requirements. As Wiktionary contains translations, this matcher can work on monolingual and multilingual matching tasks.

This is the second OAEI participation of this matching system, *Wiktionary Matcher* initially participated in the OAEI in 2019 [10]. The matcher has been implemented and packaged using the *Matching EvaLuation Toolkit (MELT)*[7], a Java framework for matcher development, tuning, evaluation, and packaging [4,9].

1.2 Specific Techniques Used

This matching system system was initially introduced at the OAEI 2019 [10]. An overview of the matching system is provided in Figure 1. The main techniques used for matching are summarized below.

Monolingual Matching For monolingual ontologies, the matching system first applies multiple string matching techniques. Afterwards, the synonym matcher module links labels to concepts in Wiktionary and checks then whether the concepts are synonymous in the external dataset. This approach is conceptually similar to an upper ontology matching approach. Concerning the usage of a collaboratively built knowledge source, the approach is similar to *WikiMatch* [3] which exploits the Wikipedia search engine. *Wiktionary Matcher* adds a correspondence to the final alignment purely based on the synonymy relation independently of the actual word sense. This is done in order to avoid word sense disambiguation on the ontology side but also on Wiktionary side: Versions for some countries do not annotate synonyms and translations for senses but rather on the level of the lemma. Hence, many synonyms are given independently of the word sense. In such cases, word-sense-disambiguation would have to be performed also on Wiktionary [8]. The linking process is similar to the one presented for the *ALOD2Vec 2018* matching system [12]: In a first step, the full label is looked up in the knowledge source. If the label cannot be found, labels consisting of multiple word tokens are truncated from the right and the process is repeated to check for sub-concepts. This allows to detect long sub-concepts even if the full string cannot be found. Label *conference banquet* of concept *http://ekaw#Conference_Banquet* from the *Conference* track, for example, cannot be linked to the background dataset using the full label. However, by applying right-to-left truncation, the label can be linked to two concepts, namely *conference* and *banquet*, and in the following also be matched to the correct concept *http://edas#ConferenceDinner* which is linked in the same fashion. For multi-linked concepts (such as *conference dinner*), a match is only annotated

[6] Namely: Dutch, English, French, Italian, German, Portugese, Russian, and Spanish.
[7] see `https://github.com/dwslab/melt`

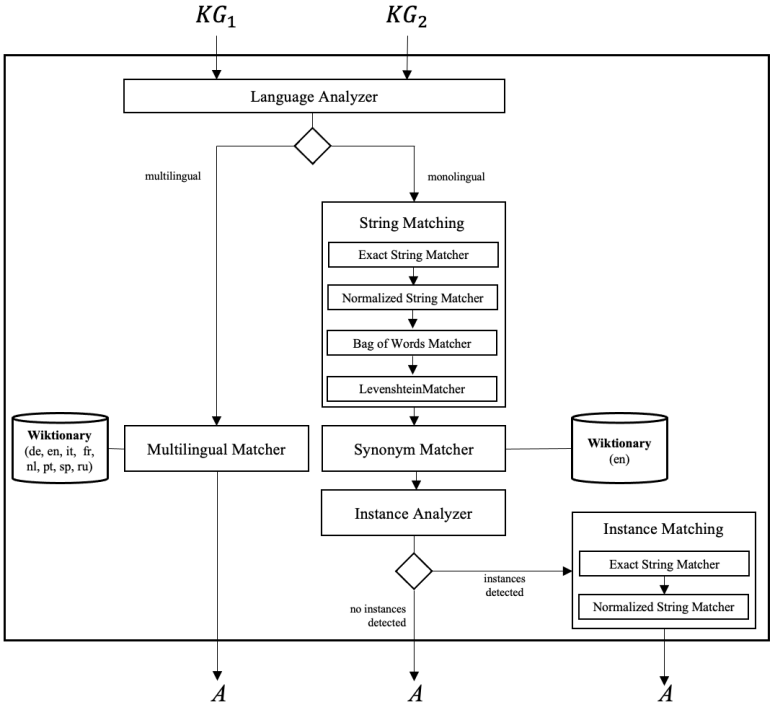


Fig. 1. High-level overview of the *Wiktionary Matcher*. KG_1 and KG_2 represent the input ontologies and optionally instances. The final alignment is referred to as A.

if every linked component of the label is synonymous to a component in the other label. Therefore, *lens (http://mouse.owl#MA_0000275)* is not mapped to *crystalline_lens (http://human.owl#NCI_C12743)* due to a missing synonymous partner for *crystalline* whereas *urinary bladder neck (http://mouse.owl#MA_0002491)* is matched to *bladder neck (http://human.owl#NCI_C12336)* because *urinary bladder* is synonymous to *bladder*.

Multilingual Matching For every matching task, the system first determines the language distributions in the ontologies. If the ontologies appear to be in different languages, the system automatically enables the multilingual matching module: Here, Wiktionary translations are exploited: A match is created, if one label can be translated to the other one according to at least one Wiktionary language version – such as the Spanish label *ciudad* and the French label *ville* (both meaning *city*). This process is depicted in Figure 2: The Spanish label is linked to the entry in the Spanish Wiktionary and from the entry the translation is derived. If there is no Wiktionary version for the languages to be matched or the approach described above yields very few results, it is checked whether the

two labels appear as a translation for the same word. The Chinese label 决定 (juédìng), for instance, is matched to the Arabic label قرار (qrār) because both appear as a translation of the English word *decision* on Wiktionary. This (less precise) approach is particularly important for language pairs for which no Wiktionary dataset is available to the matcher (such as Chinese and Arabic). The process is depicted in Figure 3: The Arabic and Chinese labels cannot be linked to Wiktionary entries but, instead, appear as translation for the same concept.

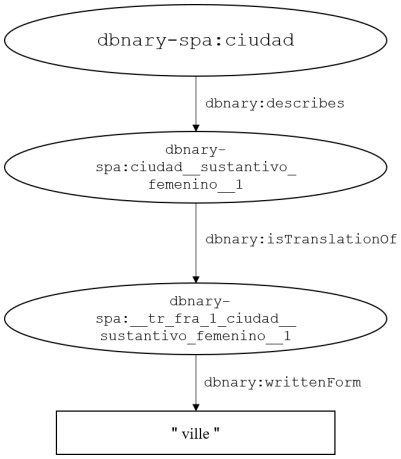


Fig. 2. Translation via the Wiktionary headword (using the DBnary RDF graph). Here: One (of more) French translations for the Spanish word *ciudad* in the Spanish Wiktionary.

Instance Matching The matcher presented in this paper can be also used for combined schema and instance matching tasks. If instances are available in the given datasets, the matcher applies a two step strategy: After aligning the schemas, instances are matched using a string index. As there are typically many instances, Wiktionary is not used for the instance matching task in order to increase the matching runtime performance. Moreover, the coverage of schema level concepts in Wiktionary is much higher than for instance level concepts: For example, there is a sophisticated representation of the concept *movie*[8], but hardly any individual movies in Wiktionary. For correspondences where the instances belong to classes that were matched before, a higher confidence is assigned. If one instance matches multiple other instances, the correspondence is preferred where both their classes were matched before.

Explainability Unlike many other ontology matchers, this matcher uses the extension capabilities of the alignment format [1] in order to provide a human

[8] see `https://en.wiktionary.org/wiki/movie`

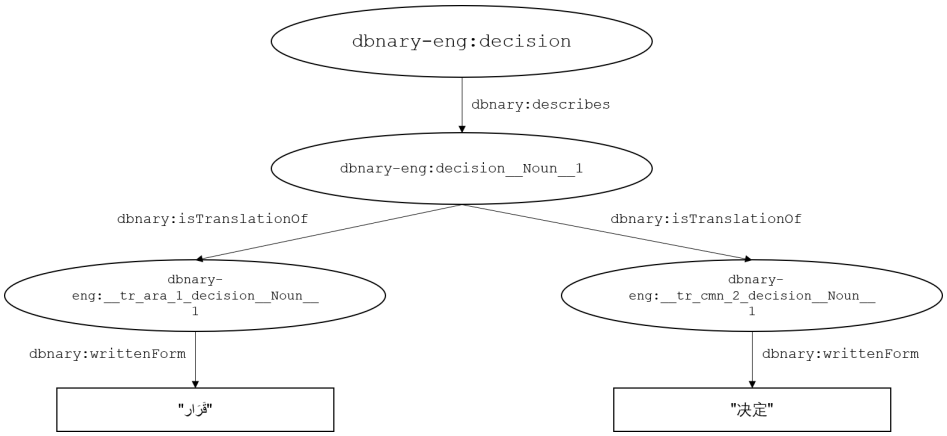


Fig. 3. Translation via the written forms of Wiktionary entries (using the DBnary RDF graph). Here: An Arabic and a Chinese label appear as translation for the same Wiktionary entry (*decision* in the English Wiktionary).

readable explanation of why a correspondence was added to the final alignment. Such explanations can help to interpret and to trust a matching system's decision. Similarly, explanations also allow to comprehend why a correspondence was falsely added to the final alignment: The explanation for the false positive match *(http://confOf#Contribution, http://iasted#Tax)*, for instance, is given as follows: *"The first concept was mapped to dictionary entry [contribution] and the second concept was mapped to dictionary entry [tax]. According to Wiktionary, those two concepts are synonymous."* Here, it can be seen that the matcher was successful in linking the labels to but failed due to the missing word sense disambiguation. In order to explain a correspondence, the `description` property[9] of the *Dublin Core Metadata Initiative* is used.

1.3 Extensions to the Matching System for the 2020 Campaign

For the 2020 campaign, the matching system has been improved. The instance matching module has been extended to better exploit the string indices. As a consequence, the matcher is the best performing system in the knowledge graph track [6] this year. Furthermore, *Wiktionary Matcher* now gives more detailed explanations in terms of why a correspondence has been added to the alignment. Lastly, the background knowledge has been updated: The system uses Wiktionary dumps as of late July 2020. The 2020 system uses the latest version of MELT [5]. The implementation is now also publicly available on GitHub.[10]

[9] see `http://purl.org/dc/terms/description`
[10] see `https://github.com/janothan/WiktionaryMatcher`

2 Results

2.1 Anatomy Track

On the anatomy track, recall and F_1 could be improved compared to the 2019 version of the matcher. Due to further improvements of the implementation, the matching system's runtime performance could be significantly increased and the system is able to align the two ontologies in less than 100 seconds.[11] The system performs above the median of all 2020 systems with an F_1 score of 0.842 (precision = 0.956, recall = 0.753).

2.2 Conference Track

The matching system achieves almost the same results as in 2019 on the conference track with a slightly improved precision. With an F_1 score of 0.65 on `rar2-M1`, the system performs slightly above the median in terms of F_1.

2.3 Multifarm Track

Wiktionary Matcher is one of the few systems capable of matching multilingual ontologies. This year, *Wikitionary Matcher* is the system with the highest precision on the aggregated results (precision = 0.8 on different ontologies). In terms of f-measure, the system scores at the exact median. Compared to the 2019 campaign, the results improved slightly. This effect is caused by the updated DBnary dataset used this year – the system improved itself due to a growing knowledge source (the multilingual matching implementation has not been changed compared to 2019).

2.4 LargeBio Track

Although the system has not been optimized for the LargeBio track, the matcher could complete all matching tasks within the given time. The system performs surprisingly competitive despite not using any other background knowledge source than Wiktionary. With the exception of task "FMA/NCI Whole", the matching system performed significantly better than the 2019 version in terms of F_1. A small contributor to better results is also the new Wiktionary version which carries more synonyms in 2020 than in 2019.

2.5 Knowledge Graph Track

Due to an improved instance matching module, the overall instance matching performance in terms of F_1 could be increased from 0.79 to 0.87. With an overall

[11] In the 2020 campaign, only 4 out of 11 systems were able to align the ontologies in less than 100 seconds.

f-measure of 0.87, *Wiktionary Matcher* is the best matching system on this track.[12]

3 General Comments

It is important to note that the matching system currently exploits only a small share of semantic relations available on Wiktionary. The system is restricted by the available relations extracted by the DBnary project. The additional exploitation of the relations *alternative forms* or *derived terms*, for instance, would likely improve the system. However, those are not yet extracted and are consequently not used for the matching task as of today.[13]

4 Conclusion

In this paper, we presented the *Wiktionary Matcher*, a matcher utilizing a collaboratively built lexical resource, as well as the results of the system in the 2020 OAEI campaign. Overall, the results of the matching system could be significantly improved compared to its last OAEI participation. Given Wiktionary's continuous growth, it can be expected that the matching results will improve over time – for example when additional synonyms and translations are added. Small improvements due to new synonyms and translations could already be observed within a one year time frame for example on the Multifarm or the LargeBio track. In addition, improvements to the DBnary dataset, such as the addition of alternative word forms, may also improve the overall matcher performance in the future.

References

1. David, J., Euzenat, J., Scharffe, F., dos Santos, C.T.: The alignment API 4.0. Semantic Web **2**(1), 3–10 (2011). https://doi.org/10.3233/SW-2011-0028, `https://doi.org/10.3233/SW-2011-0028`
2. Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. Language, Speech, and Communication, MIT Press, Cambridge, Massachusetts (1998)
3. Hertling, S., Paulheim, H.: WikiMatch - Using Wikipedia for Ontology Matching. In: Shvaiko, P., Euzenat, J., Kementsietsidis, A., Mao, M., Noy, N., Stuckenschmidt, H. (eds.) OM-2012: Proceedings of the ISWC Workshop. vol. 946, pp. 37–48 (2012)

[12] *ALOD2Vec Matcher 2020* [11] achieves the same F_1 score – however, as the performance of the latter matcher on classes and properties is slightly worse, *Wiktionary Matcher* comes in first.

[13] We contacted the developers and will include the additional relations in our matching system as soon as those are available.

4. Hertling, S., Portisch, J., Paulheim, H.: MELT - matching evaluation toolkit. In: Acosta, M., Cudré-Mauroux, P., Maleshkova, M., Pellegrini, T., Sack, H., Sure-Vetter, Y. (eds.) Semantic Systems. The Power of AI and Knowledge Graphs - 15th International Conference, SEMANTiCS 2019, Karlsruhe, Germany, September 9-12, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11702, pp. 231–245. Springer (2019). https://doi.org/10.1007/978-3-030-33220-4_17, https://doi.org/10.1007/978-3-030-33220-4_17
5. Hertling, S., Portisch, J., Paulheim, H.: Supervised ontology and instance matching with MELT. In: OM@ISWC 2020 (2020), to appear
6. Hofmann, A., Perchani, S., Portisch, J., Hertling, S., Paulheim, H.: Dbkwik: Towards knowledge graph creation from thousands of wikis. In: Proceedings of the ISWC 2017 Posters & Demonstrations and Industry Tracks co-located with 16th International Semantic Web Conference (ISWC 2017), Vienna, Austria, October 23rd - to - 25th, 2017 (2017)
7. McCrae, J., Aguado-de Cea, G., Buitelaar, P., Cimiano, P., Declerck, T., Gómez-Pérez, A., Gracia, J., Hollink, L., Montiel-Ponsoda, E., Spohr, D., Wunner, T.: Interchanging Lexical Resources on the Semantic Web. Language Resources and Evaluation **46**(4), 701–719 (Dec 2012). https://doi.org/10.1007/s10579-012-9182-3, http://link.springer.com/10.1007/s10579-012-9182-3
8. Meyer, C.M., Gurevych, I.: Worth its weight in gold or yet another resource - A comparative study of wiktionary, openthesaurus and germanet. In: Gelbukh, A.F. (ed.) Computational Linguistics and Intelligent Text Processing, 11th International Conference, CICLing 2010, Iasi, Romania, March 21-27, 2010. Proceedings. Lecture Notes in Computer Science, vol. 6008, pp. 38–49. Springer (2010). https://doi.org/10.1007/978-3-642-12116-6_4, https://doi.org/10.1007/978-3-642-12116-6_4
9. Portisch, J., Hertling, S., Paulheim, H.: Visual analysis of ontology matching results with the MELT dashboard. In: The Semantic Web: ESWC 2020 Satellite Events (2020)
10. Portisch, J., Hladik, M., Paulheim, H.: Wiktionary matcher. In: Shvaiko, P., Euzenat, J., Jiménez-Ruiz, E., Hassanzadeh, O., Trojahn, C. (eds.) Proceedings of the 14th International Workshop on Ontology Matching co-located with the 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26, 2019. CEUR Workshop Proceedings, vol. 2536, pp. 181–188. CEUR-WS.org (2019), http://ceur-ws.org/Vol-2536/oaei19_paper15.pdf
11. Portisch, J., Hladik, M., Paulheim, H.: ALOD2Vec Matcher results for OAEI 2020. In: OM@ISWC 2020 (2020), to appear
12. Portisch, J., Paulheim, H.: Alod2vec matcher. In: OM@ISWC. CEUR Workshop Proceedings, vol. 2288, pp. 132–137. CEUR-WS.org (2018)
13. Sérasset, G.: Dbnary: Wiktionary as a lemon-based multilingual lexical resource in RDF. Semantic Web **6**(4), 355–361 (2015). https://doi.org/10.3233/SW-140147, https://doi.org/10.3233/SW-140147

Ontology Alignment in Ecotoxicological Effect Prediction*

Erik B. Myklebust[1,2], Ernesto Jiménez-Ruiz[2,3], Jiaoyan Chen[4], Raoul Wolf[1], and Knut Erik Tollefsen[1,5]

[1] Norwegian Institute for Water Research, Oslo, Norway
[2] SIRIUS, University of Oslo, Oslo, Norway
[3] City, University of London, London, United Kingdom
[4] University of Oxford, Oxford, United Kingdom
[5] Norwegian University of Life Sciences, Ås, Norway

1 Introduction

The Toxicological and Risk Assessment Knowledge Graph (TERA) [1] integrates several disparate datasets relevant to ecological risk assessment and effect prediction. TERA is being used in conjunction with knowledge graph embedding models to improve the extrapolation of chemical effect data in the Norwegian Institute for Water Research (Norsk institutt for vannforskning, NIVA) [1].[1]

The largest publicly available repository of effect data is the ECOTOXicology knowledge base (ECOTOX) developed by the US Environmental Protection Agency [2]. The dataset consists of $940k$ experiments using $12k$ compounds and $13k$ species. ECOTOX contains a taxonomy (of species), however, this only considers the species represented in the ECOTOX effect data. Hence, to enable extrapolation of effects across a larger taxonomic domain, an alignment to the NCBI taxonomy have to be established. However, there does not exist a complete and public mapping set between the 47,785 ECOTOX taxa and the 2,140,344 NCBI taxa. In this paper we present the ECOTOX-NCBI alignment results of three ontology matching algorithms.

2 Methods and Evaluation

Although there does not exist a complete and public alignment between the ECOTOX and NCBI, a partial mapping curated by experts can be obtained through the ECOTOX Web.[2] We have gathered a total of 2,321 mappings for validation purposes. We have used three methods to align the two vocabularies: *(i)* LogMap system [3]. *(ii)* AgreementMakerLight (AML) , and *(iii)* a baseline string matching algorithm based on Levenshtein distance [4].

Table 1 shows the alignment results over the ground truth samples. Note that the results represent 1-to-1 alignments as, in our setting, it is expected an entity from

[1] Knowledge Graphs at NIVA: `https://github.com/NIVA-Knowledge-Graph/`

[2] ECOTOX search interface: `https://cfpub.epa.gov/ecotox/search.cfm`

Algorithm	# mappings	Recall	Precision (*)
LogMap	32,726	0.81	0.88
AML	31,659	0.80	0.87
String distance (> 0.8)	33,554	0.38	0.70
Union all	57,511	0.72	0.73
Consensus (LogMap $\cap$ AML)	20,217	0.78	0.95
LogMap $\cup$ AML	39,985	0.83	0.85

Table 1. Alignment results for ECOTOX-NCBI. (*) Estimated precision with respect to the known entities in the incomplete reference alignment, assuming only 1-1 mappings are valid.

ECOTOX to match to a single entity in NCBI, and vice-versa. Hence, 1-to-N (respectively N-to-1) alignments were filtered according to the system computed confidence. LogMap and AML produce mapping sets with similar recall and (estimated) precision, with LogMap producing a larger number of mappings. The baseline matcher, as expected, achieves both a lower recall and (estimated) precision. This shows that a simple string matching solution may not be enough in this setting. Table 1 also shows the results of the consensus alignment between AML and LogMap and the union of different mapping sets. Note that the lower recall of the union is down to overconfidence in the string distance method when 1-to-1 filtering.

3 Conclusions

The used alignment techniques achieve relatively good scores for recall over the available (incomplete) reference mappings. However, aligning such large and challenging datasets required some preprocessing before ontology alignment systems could cope with them. The preprocessing involved to split NCBI into manageable fragments, leading to a set of matching subtasks instead of a single task. Thus, the alignment of ECOTOX and NCBI has the potential of becoming a new track of the Ontology Alignment Evaluation Initiative (OAEI)[3] [5] to push the limits of state-of-the-art systems. The output of the different OAEI participants could be merged into a rich consensus alignment that could become the reference to integrate ECOTOX and NCBI. At the same time, as the alignment between ECOTOX and NCBI is not public nor complete, the consensus mappings could also be seen as a very relevant resource to the ecotoxicology community.

References

1. Myklebust, E.B., Jiménez-Ruiz, E., Chen, J., Wolf, R., Tollefsen, K.E.: Knowledge Graph Embedding for Ecotoxicological Effect Prediction. In: ISWC. (2019)
2. U.S. EPA: ECOTOXicology knowledgebase (ECOTOX) (2019)
3. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: ECAI. (2012)
4. Levenshtein, V.I.: Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Soviet Physics Doklady **10** (1966)
5. Algergawy, A., et al.: Results of the Ontology Alignment Evaluation Initiative 2019. In: 14th International Workshop on Ontology Matching. (2019) 46–85

[3] OAEI: `http://oaei.ontologymatching.org/`

Towards Semantic Alignment of Heterogeneous Structures and Its Application to Digital Humanities

Renata Vieira[1]* and Cassia Trojahn[2]

[1] CIDEHUS, University of Évora, Portugal
renatav@uevora.pt
[2] IRIT, UMR 5505, 1118 Route de Narbonne, F-31062 Toulouse, France
firstname.lastname@irit.fr

1 Introduction

The field of Digital Humanities comprises the use of technology within arts, heritage and humanities research. This brings new methods of inquiry, new means of dissemination, but also constitute a new core of investigation in itself. Not only creation and access to collections of interest for these areas have improved with digitalization of research material, but further use of computing technology is being proposed and discovered [7]. The primary source of information for humanities researchers comes from free, unstructured sources in written language, that is ambiguous and context-dependent. Also the humanities might face difficulties due to the particularities of the source of information, that might be available in ancient forms of registration. For instance, there is a need for identifying specific vocabulary of a historical period and also align non uniform spelling which was usual in old publications [6]. In this perspective, the ability to establish a relationship between different forms of expression of knowledge (from structured and unstructured sources) and its meaning or intent is crucial [5]. This scenario reflects a unifying framework of a wide range of solutions from a variety of domains, including NLP and semantic web.

Different variants of the notion of 'alignment' have been adopted in a range of areas, focusing on homogeneous structures (e.g., text alignment [8], database alignment [1] or ontology alignment [4]) or heterogeneous structures (e.g., annotation of text with ontologies [3], alignment of dictionaries and ontologies [2], alignments between relational databases and ontologies [9]). These alignment approaches, however, take little account of the alignment of multiple structures. This type of approach is becoming increasingly necessary to manage the growing volume of unstructured information sources available on the Web (encyclopedias such as Wikipedia, social media data, etc.) and LOD knowledge bases. In addition, the approaches are mostly developed for the English language. These needs have to be addressed through a global vision of alignment that takes into account a multiplicity of structures in which knowledge can be expressed. This paper seeks a holistic approach to semantic computing and alignment, when considering heterogeneous structures in which knowledge is represented.

2 Proposal

The approach consists of two main steps. First, knowledge extraction approaches will be applied to extract the terminology of the relevant corpora. We plan to specialise general language models, since the corpora present distinctive language characteristics due to scope and time. We also plan to make use of techniques for the recognition of named entities which might help finding important relations and events. On the basis of the models and recognised entities we plan to extract other information with the help of semantic alignment methods. Second, the extracted terminology will be aligned to existing sources of knowledge (available dictionaries, lexicons, corpora and ontologies). In particular, there are basic ontological concepts describing fundamental elements such as persons, places, periods, and that have to be anchored to what is extracted. Ontologies will be the central focus for semantic alignment of textual occurrences of concepts, and its relations with other semantic sources. The alignment may consider previous semantic knowledge, or might be inferred trough semantic similarity analysis.

We plan to apply our approach on current projects such as the Curvo Semedo's works [6]. This is a corpus integrated by six works published between 1707 and 1727, authored by Alentejo doctor João Curvo Semedo (1635-1719), containing medical and pharmacological knowledge constituted and published in Portuguese. The focus reader of his works, at the time they were recorded, was a less educated person, little affected by the materials available only in Latin. The six works gathered include a collection of about 2,150 pages, which are treated and offered in the form of transcripts, in different formats, in original spelling and reproduced, accompanied by descriptions of their terminologies and representations of the content of each one, generated with the support of computational tools. The evaluation phase will be carried out with he help of humanities expert. The proposed methodology has potential utility for other projects with a variety of history and linguistic inquiries.

References

1. J. Cole, Q. Wang, et al. The ribosomal database project: improved alignments and new tools for rrna analysis. *Nucleic acids research*, 37:D141–D145, 2009.
2. B. Dalvi, E. Minkov, P. P. Talukdar, and W. W. Cohen. Automatic gloss finding for a knowledge base using ontological constraints. In *8th Conf. WSDM*, pages 369–378, 2015.
3. M. Erdmann, A. Maedche, H.-P. Schnurr, and S. Staab. From manual to semi-automatic semantic annotation. In *COLING Workshop on Semantic Annotation*, pages 79–85, 2000.
4. J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg, Germany, 2007.
5. M. Matuschek and I. Gurevych. Dijkstra-wsa: A graph-based approach to word sense alignment. *TACL*, 1:151–164, 2013.
6. P. Quaresma and M. J. B. Finatto. Information extraction from historical texts: a case study. In *DHandNLP@PROPOR*, pages 49–56, 2020.
7. S. Schreibman, R. Siemens, and J. Unsworth. *A new companion to digital humanities*. John Wiley & Sons, 2015.
8. D. Tufiş, A. M. Barbu, and R. Ion. Extracting multilingual lexicons from parallel corpora. *Computers and the Humanities*, 38(2):163–189, 2004.
9. D. Uña, N. Rümmele, G. Gange, et al. Machine learning and constraint programming for relational-to-ontology schema mapping. In *27th IJCAI*, pages 1277–1283, 2018.

Ontology Matching for the Laboratory Analytics Domain*

Ian Harrow[1], Thomas Liener[1], and Ernesto Jiménez-Ruiz[2,3]

[1] Ontologies Mapping Project, Pistoia Alliance, USA
[2] City, University of London, United Kingdom
[3] SIRIUS, Department of Informatics, University of Oslo, Norway

1 Introduction

The Pistoia Alliance was established ten years ago to promote innovation by industry through pre-competitive collaboration to reduce the barriers to innovation. The Ontologies Mapping Project started in 2016 to enable better tools and services for ontology mapping and to define best practices for ontology management in the Life Sciences [1].

The interest in ontologies is growing within the pharmaceutical domain. Data is a very valuable corporate asset to enable digital transformation and lead to innovative biological insight. However, data integration is fundamental piece in the puzzle where ontologies and ontology matching may play an important role.

The Pistoia Alliance Ontologies Mapping Project has covered two domains of interest: *(i)* phenotype and disease [2], and *(ii)* laboratory analytics domain. In this paper we focus on the later, for which alignment sets are not that common, we introduce the system Paxo, and we compare its results against participants of the Ontology Alignment Evaluation Initiative (OAEI, `http://oaei.ontologymatching.org/`).

Datasets. We selected, in conjunction with (pharmaceutical) industry partners of the Pistoia Alliance, 9 relevant ontologies to the laboratory analytics domain and 13 ontology pairs to compute their alignment. Table 1 shows the ontologies that were selected for their relevance to the laboratory analytics domain. Note that there is not a public hand-curated gold standard alignment among the selected ontology pairs.

Paxo system. Paxo is a lightweight ontology mapping approach. Unlike other algorithms, Paxo does not need to store, load or index ontologies. Instead Paxo accesses the API of the Ontology Lookup Service (OLS, `https://www.ebi.ac.uk/ols/index`) and the Ontology Mapping Repository (OxO, `https://www.ebi.ac.uk/spot/oxo/`) at EMBL-EBI to explore ontologies. Through OLS, Paxo can perform search via preferred label and synonyms, while OxO offers access to a wide range of known ontology mappings, that were defined, for example, as cross references within the ontologies themselves or in the UMLS Metathesaurus.

2 Evaluation

Table 2 shows the number of computed mappings, for the 13 selected matching tasks, by Paxo (with relaxed-R and strict-S variants) and a subset of the OAEI systems that were able to cope with (most of) the selected matching tasks.

We have computed consensus alignments of vote 2, 3 and 4 (*i.e.*, mappings suggested by at least two, three or four systems, respectively). Note that, when there are several systems of the same family (*i.e.*, systems participating with several variants), their (voted) mappings are only counted once in order to reduce bias.

Domain	Ontology Name	Acronym	Size	Version
Chemistry	Allotrope Merged Ontology Suite	AFO	1,868	2019/05/10
	Chemical Methods Ontology	CHMO	3,130	2014-11-20
Biology	Ontology for Biomedical Investigations	OBI	3,959	2019-11-12
	Eagle-I Research Resource Ontology	ERO	4,334	23-07-2019
	Mass Spectrometry Ontology	MS	6,855	19:11:2019
	BioAssay Ontology	BAO	7,512	2.5.1
	Experimentals Factors Ontology	EFO	26,510	3.12.0
General	National Cancer Institute Thesaurus	NCIT	154,108	19.11d
	Medical Subject Headings	MESH	539,242	2019ab

Table 1: Ontologies relevant to the laboratory analytics domain.

Matching Task	System mappings						Consensus mappings			
	Paxo-R	Paxo-S	AML	BioPortal	LogMap	LogMapBio	#SF	Con-2	Con-3	Con-4
AFO-CHMO	234	199	214	160	240	247	6	220	200	176
AFO-MESH	149	76	130	39	152	153	4	120	57	32
AFO-NCIT	461	313	361	213	297	315	4	403	224	159
BAO-MESH	273	176	248	112	313	317	4	251	142	81
BAO-NCIT	564	418	249	230	232	250	6	304	255	242
CHMO-MESH	435	222	240	70	252	257	4	229	124	62
CHMO-NCIT	605	343	196	125	171	209	7	215	151	128
EFO-MESH	3,710	2,953	3,392	1,250	3,054	3,344	4	3,140	2,538	1,170
EFO-NCIT	4,297	3,559	(-)	2,442	3,448	4,047	4	3,054	2,477	2,266
ERO-MESH	277	176	165	74	206	205	4	174	120	65
ERO-NCIT	511	343	174	168	168	194	7	234	191	177
MS-NCIT	268	143	73	86	56	57	5	107	86	74
OBI-NCIT	504	302	137	147	142	155	7	186	155	149

Table 2: Number of mappings for the selected matching tasks. (-): a system failed to compute mappings. #SF: number of system families contributing to the consensus. Con-x: consensus mappings with 'x' votes. We focus on the entities defined in the input ontologies and thus ignore entities imported/reused from external ontologies.

Paxo-R is the system that, on average, predicts the highest amount of mappings followed by LogMap-Bio and Paxo-S; while BioPortal includes, on average, the smallest amount. Figure 1 shows a two-dimensional representation of the *Jaccard distances* among the alignments between EFO and MESH. Paxo-R and Paxo-S produce relatively similar mapping sets (as for LogMap and LogMap-Bio). Being close to a consensus mapping set is not necessarily positive; but it means that the computed mappings are similar to the agreement. For example, BioPortal mappings are typically small in size and close to Con-4. Paxo mappings are different from the other system computed mappings. A more detailed (manual) analysis will be conducted in the near future to evaluate the quality of the reported mapping sets.

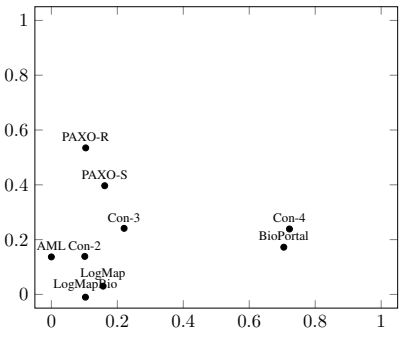


Fig. 1: Two-dimensional representation of the Jaccard distances among EFO-MESH mappings. Plots computed with the MELT framework (`https://github.com/dwslab/melt`).

References

1. Harrow, I., et al.: Ontology mapping for semantically enabled applications. Drug Discovery Today (2019)
2. Harrow, I., Jiménez-Ruiz, E., et al.: Matching disease and phenotype ontologies in the ontology alignment evaluation initiative. J. Biomedical Semantics **8**(1) (2017)

Towards Matching of Domain Ontologies to Cross-Domain Ontology: Evaluation Perspective

Martin Šatra and Ondřej Zamazal

Department of Information and Knowledge Engineering,
University of Economics, W. Churchill Sq.4, 130 67 Prague 3, Czech Republic,
{satm03|ondrej.zamazal}@vse.cz

1 Introduction

Ontology matching, as a process of matching two or more ontologies, is usually aimed at matching of domain ontologies. However, there are also other kinds of ontologies which make sense to align (and particularly with domain ontologies). Cross-domain (general) ontologies cover more domains. For example, the *DBpedia ontology* is a cross-domain ontology. It contains concepts, such as *Agent*, *Device*, *Food*, *Place*, from diverse domains. In comparison, domain ontologies focus on concepts from one area. For instance, the *confof* ontology from OntoFarm[1] contains concepts such as *Contribution*, *Event*, *Person* dealing with the conference organization.

While motivation use cases (such as *information integration* and *information sharing*, e.g. in [1]) for matching of domain ontologies to a cross-domain ontology are to a large degree similar as for matching of domain ontologies, there are different challenges with regard to matching. We claim that matching to cross-domain ontology is more difficult for traditional ontology matching systems since a cross-domain ontology contains concepts from various areas and it is more difficult to recognize proper concepts to align. Next a cross-domain ontology is usually larger. In all, we can expect a higher amount of false positives (lowering precision) since string-based matching techniques will be more often confused. There has not yet been much work done on this kind of matching. Authors in [3] focused on matching enhanced with knowledge of the domain and they evaluated their approach on matching two domain ontologies to the DBpedia ontology. Further there is a close effort of matching of foundational ontologies [2].

2 Reference Alignment and Evaluation

For building of reference alignments (RA) we merely focused on entities of *DBpedia* ontology [2] from DBpedia namespace and three ontologies from OntoFarm:

[1] https://owl.vse.cz/ontofarm/

[2] http://downloads.dbpedia.org/2016-10/dbpedia_2016-10.owl

confof, *ekaw*, *sigkdd*. The process of constructing RA was supported by basic ontology matching techniques available from the Alignment API.[3] Further, a thorough manual matching was applied. Based on these input a tentative RA were prepared.[4] Finally, the RA were reconciled with the existing RA for the conference track of OAEI (Ontology Alignment Evaluation Initiative)[5] consisting of correspondences between OntoFarm domain ontologies. The resulted RA contain both equivalence and subsumption correspondences with 1:1 cardinality.[6]

For evaluation (merely equivalence correspondences) we employed several matching systems from OAEI 2019: *AML*, *DOME*, *LogMap* and *LogMapLt*.[7] According to the results in Table 1 AML, DOME and LogMap have very simlar results in terms of F_1-measure. While LogMap is better in precision, AML and DOME are better in recall. The system based only on string technique, LogMapLt, has the lowest F_1-measure. As expected evaluation metrics are rather low (e.g. 0.42 vs. 0.70 in terms of comparing F_1-measures with regard to the result of matching of domain ontologies in the conference track of OAEI 2019).

Table 1. Precision, F_1-measure and Recall for systems (micro-average).

System	Prec.	F_1-m.	Rec.
AML	0.30	**0.42**	0.67
DOME	0.32	**0.42**	0.60
LogMap	0.37	0.41	0.47
LogMapLt	0.33	0.36	0.40

3 Conclusions and Future Work

Low scores of measures show that the corresponding test cases are difficult for traditional ontology matching systems since they mainly focus on matching of domain ontologies. In future we plan to engage more systems and we also plan to extend the RA. We envisage to employ the RA within the conference track of the OAEI 2020 as a new challenge for matching systems.

Ondřej Zamazal is supported by the CSF grant no. 18-23964S.

References

1. G. Kobilarov, T. Scott, Y. Raimond, S. Oliver, C. Sizemore, M. Smethurst, C. Bizer, and R. Lee. Media meets semantic web–how the bbc uses dbpedia and linked data to make connections. In *ESWC*. Springer, 2009.
2. D. Schmidt, A. Pease, C. Trojahn, and R. Vieira. Aligning conference ontologies with SUMO: A report on manual alignment via wordnet. In *Proc. of the Joint Ontology Workshops*, CEUR, 2019.
3. K. Slabbekoorn, L. Hollink, and G.-J. Houben. Domain-aware ontology matching. In *International Semantic Web Conference*, pages 542–558. Springer, 2012.

[3] `http://alignapi.gforge.inria.fr/`
[4] RA were done by one evaluator and eventually one referee confirmed the resulted RA during a discussion.
[5] `http://oaei.ontologymatching.org/`
[6] Available on the OntoFarm web, `https://owl.vse.cz/ontofarm/#ra-to-dbpedia`.
[7] System papers are available at `http://om2019.ontologymatching.org/#ap`

Towards a Vocabulary for Mapping Quality Assessment

Alex Randles [0000-0001-6231-3801], Ademar Crotti Junior [0000-0003-1025-9262] and Declan O'Sullivan [0000-0003-1090-3548]

ADAPT Centre, Trinity College Dublin, Dublin 2, Ireland
{alex.randles,ademar.crotti,declan.osullivan}@adaptcentre.ie

Abstract. This paper presents a vocabulary for expressing information related to the assessment, refinement and validation of mappings called the Mapping Quality Vocabulary.

1 Introduction

Oftentimes, RDF datasets are generated by converting non-RDF resources to RDF in a process called 'uplift'. The uplift process commonly involves the definition of mappings, which allows one to declaratively express the transformations needed to convert non-RDF source data to RDF [1]. Another use for mappings is found when relating and interlinking those RDF datasets. The creation of such mapping definitions is a complex time-consuming task, involving various quality related activities in which mappings are iteratively refined until they satisfy its stakeholders expressed requirements. While approaches for assessing the quality of mappings have been proposed [2, 3], a vocabulary for describing such processes is still lacking. To tackle this problem, we present the Mapping Quality Vocabulary (MQV) which aims at enabling quality metadata and provenance information relating to the assessment and refinement of mappings to be captured and published.

2 Mapping Quality Vocabulary

The proposed Mapping Quality Vocabulary[1] provides a vocabulary for expressing information relating to the quality assessment, refinement and validation of mappings. The goal is to make this information easier to publish, exchange and consume. Our proposed model is separated into three stages: **assessment**, **refinement** and **validation**. **Fig. 1** provides a general overview of the core components of the MQV model.

Mapping quality assessment. In this stage, one or more mapping documents (`mqv:MappingDocument`) are assessed. An assessment activity is captured through the `mqv:MappingAssessment` class. A mapping assessment activity may have

[1] Mapping Quality Vocabulary Specification available at `https://alex-randles.github.io/MQV/`

quality requirements, which are captured through the `mqv:QualityRequirement` class. We declaratively capture such information as we foresee and allow for quality validation activities to be executed in later stages of the process. The model uses this information to generate a mapping validation report (`mqv:MappingValidationReport`). Each violation identified is then represented with the `mqv:MappingViolation` class.

Mapping quality refinement. This stage involves capturing mapping refinements which are executed on the mapping to remove quality violations. Each metric described using our model may have multiple refinements (`mqv:MappingRefinement`) depending on the quality aspect being measured. The refinement executed in the mapping is associated with the identified violation through the property `mqv:wasRefinedBy`.

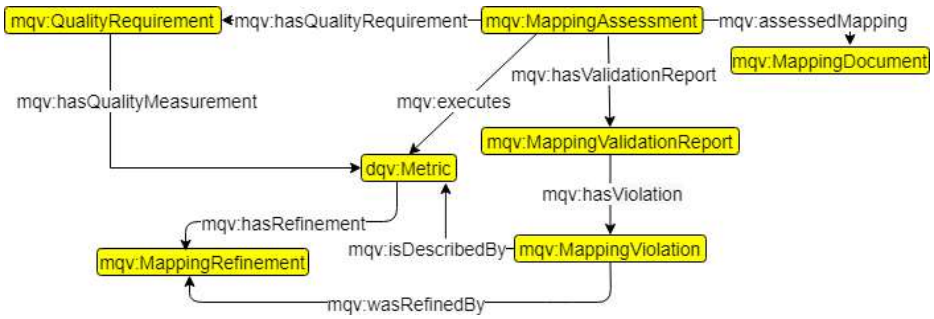


Fig. 1. Core components of the MQV model.

Mapping quality validation. Finally, our model provides quality information on the original mapping being assessed, and the mapping which has been refined in the process. As mentioned, each mapping assessment process may have quality requirements which can be validated at this stage.

3 Conclusion

The MQV model allows mapping quality information to be captured in a machine-readable format which allows it to be easily interpreted and processed by agents. Publishing this information as metadata is expected to improve the trustworthiness of the dataset, as well as encouraging the reuse and maintenance of those mappings.

References

1. Crotti, A. et al.: An evaluation of uplift mapping languages. Int. J. Web Inf. Syst. 13, 4, 405–424 (2017). https://doi.org/10.1108/IJWIS-04-2017-0036.
2. Junior, A.C. et al.: Assessing the Quality of R2RML Mappings. In: SEM4TRAAMAR@SEMANTICS. (2019).
3. Moreau, B., Serrano-Alvarado, P.: Assessing the Quality of RDF Mappings with EvaMap. In: 17th Extended Semantic Web Conference. (2020).

TableCNN: Deep Learning Framework for Learning Tabular Data *

Pranav Sankhe, Elham Khabiri, Bhavna Agrawal, and Yingjie Li

[1] University at Buffalo, Buffalo, NY
[2] IBM Research, Yorktown Heights, USA
pranavgi@buffalo.edu
{khabiri,bhavna,yingjie}@us.ibm.com

Abstract. Databases and tabular data are among the most common and rapidly growing resources. But many of these are poorly annotated (lack sufficient metadata), and are filled with domain specific jargon and alpha-numeric codes. Because of the domain specific jargon, no pre-trained language model could be applied readily to encode the cell content. We propose a deep learning based framework, TableCNN, that encodes the semantics of the surrounding cells to predict the meaning of the columns. We propose application of Byte Pair Encoding (BPE)[5] to create tokens for each cell and treat each cell as a phrase of existing tokens. Once tokenized, we process it with a CNN network to develop a classifier.

1 Introduction

Tables are rich in data and can provide vital information about the object due to the virtue of its structure. Extracting useful insights from tabular data may require domain expertise, especially if the information is comprised of domain specific jargon's or alpha-numeric codes. Our method provides a supervised learning solution to classify an unknown column in such tables into predefined column classes which can also come from a knowledge graph. Existing methods[2, 3, 1] cannot accommodate such data.

2 Methodology and Results

Cell entries are tokenized using Byte-Pair Encoding with a stopping condition defined by the token frequency threshold. Cell embedding is generated using Word2Vec[4]; each row across the tokenized table is treated as a sentence for Word2Vec model learning. We extract micro tables from the table, with a target column and surrounding columns having set number of rows; which are model parameters. Micro tables are then processed through TableCNN to classify which

* Supported by IBM Research

class it belongs to. Class is defined as the header of a column in the table used for training.

Network: TableCNN Fig 1 is an abstract view of the TableCNN network architecture. We extract column and row features in separate networks and combine them to regress final output. Row and column features are computed by a convolution operation over the first row and the target column of micro table respectively. Outputs from row and column features are concatenated and fed to a fully connected layer with a SoftMax layer to make final prediction.

For our experiment, we use a manufacturing database table that contains 112 columns and 115 thousand rows. Fig 2 shows that we obtain correct column predictions, except for column 49. Upon further inspection, we found that most of the cells in column 49 were empty, and thus the loss of classification accuracy. This shows that network is able to learn features from the surrounding columns along with the target column entries.

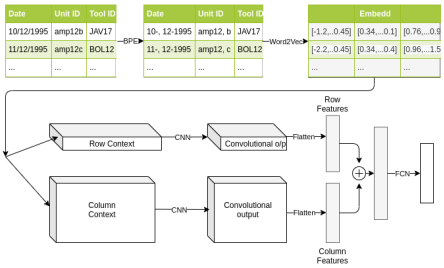


Fig. 1: TableCNN Architecture.

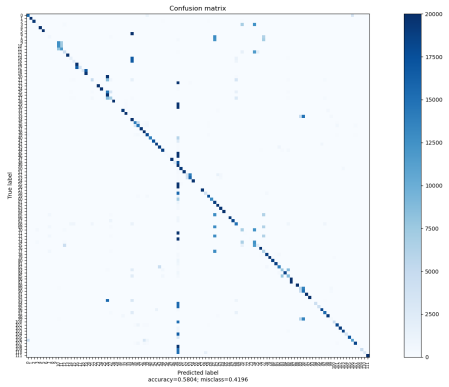


Fig. 2: Confusion Matrix.

3 Conclusions

In this poster, we present a supervised learning framework that can classify columns of a table with arbitrary alpha-numeric data. The arbitrary alpha-numeric nature of data prevents us from using pre-trained language models.

References

1. Chen, J., Jiménez-Ruiz, E., Horrocks, I., Sutton, C.A.: Learning semantic annotations for tabular data. CoRR **abs/1906.00781** (2019)
2. Efthymiou, V., Hassanzadeh, O., Rodriguez-Muro, M., Christophides, V.: Matching web tables with knowledge base entities: From entity lookups to entity embeddings. In: International Semantic Web Conference (2017)
3. Limaye, G., Sarawagi, S., Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. PVLDB **3**, 1338–1347 (09 2010)
4. Mikolov, T., Chen, K., Corrado, G.S., Dean, J.: Efficient estimation of word representations in vector space. CoRR **abs/1301.3781** (2013)
5. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. ArXiv **abs/1508.07909** (2016)