Bridging Computational Lexicography and Corpus Linguistics: A Query Extension for OntoLex-FrAC

Christian Chiarcos, Ranka Stanković, Maxim Ionov, Gilles Serasset

Bridging Computational Lexicography and Corpus Linguistics: A Query Extension for OntoLex-FrAC

Christian Chiarcos[1], **Ranka Stanković**[2], **Maxim Ionov**[3], **Gilles Sérasset**[4]
[1]Applied Computational Linguistics, University of Augsburg, Germany, christian.chiarcos@uni-a.de
[2]University of Belgrade, Faculty of Mining and Geology, Serbia, ranka.stankovic@rgf.bg.ac.rs
[3]University of Cologne, Cologne Center for eHumanities, Germany, mionov@uni-koeln.de,
[4]Université Grenoble Alpes, France, gilles.serasset@imag.fr

Abstract

OntoLex, the dominant community standard for machine-readable lexical resources in the context of RDF, Linked Data and Semantic Web technologies, is currently extended with a designated module for Frequency, Attestations and Corpus-based Information (OntoLex-FrAC). We propose a novel component for OntoLex-FrAC, addressing the incorporation of corpus queries for (a) linking dictionaries with corpus engines, (b) enabling RDF-based web services to exchange corpus queries and responses data dynamically, and (c) using conventional query languages to formalize the internal structure of collocations, word sketches, and colligations. The primary field of application of the query extension is in digital lexicography and corpus linguistics, and we present a proof-of-principle implementation in backend components of a novel platform designed to support digital lexicography for the Serbian language.

Keywords: standardization, digital lexicography, OntoLex, corpus querying, Linked Data, Linguistic Linked Open Data

1. Motivation

The exploration of digital corpora is one of the most established applications of computational approaches to linguistics and lexicography, and there are plenty of corpus management systems to support corpus-based research. Although each of these provides their own ways to access it, there are only limited efforts on standardizing the way data can be requested and exchanged. A notable exception is the CLARIN Federated Content Search (Körner et al., 2023) that allows external services to query lexical resources in a federated way – but only addressing basic morphosyntactic queries and not including advanced corpus management functionalities such as collocation search, creating word sketches, etc. A vocabulary for such requests and responses would increase inter-corpora interoperability and create a possibility for new federated applications.

We propose such vocabulary, based on web technologies such as URIs and RDF, and the OntoLex vocabulary (McCrae et al., 2017) that became the *de facto* standard for the representation of lexical resources on the Web of Data. With an addition to the emerging OntoLex module for Frequency, Attestation and Corpus-based Information (OntoLex-FrAC, or, briefly, FrAC), we aim to provide a way to encode the interaction with a corpus management system by means of API calls or HTTP GET requests; however, standardising query languages is beyond the scope of OntoLex, as this is currently pursued in other, complementary initiatives (Evert et al., 2020). Instead, we provide a minimalistic extension to OntoLex-FrAC where query expressions are treated as atomic strings, and their interpretation is guided by reference to an external URI that provides a schema, informal descriptions or other forms of documentation, as well, as – optionally –, by (human-readable) information found in an accompanying `dct:description` property. Ultimately, the goal is to facilitate the portability of systems for corpus-based lexicography over different kinds of backends, to improve the transparency and interpretability of the necessary interface representations on grounds of community standards and RDF semantics, and to more easily integrate existing lexical data with the possibility to retrieve frequency, attestations and corpus information dynamically from one or more remote corpus management systems.

We also present a proof-of-principle implementation of such a system by components for a novel platform designed to support digital lexicography for the Serbian language.

2. OntoLex-FrAC

Linguistic Linked Open Data (Cimiano et al., 2020, LLOD) is a set of best practices facilitating the sharing and reuse of linguistic data in various applications and research domains on the basis of web technologies. It is an adaptation of the Linked Open Data principles formulated for the Web of Data (Bizer et al., 2023) and implements the FAIR principles (Wilkinson et al., 2016) not just on the level of metadata, but also on the level of data (Cimiano et al., 2020, p.4-9). Based on web standards, the HTTP protocol, resolvable URIs and the adherence to open standards (be it for open or closed-source

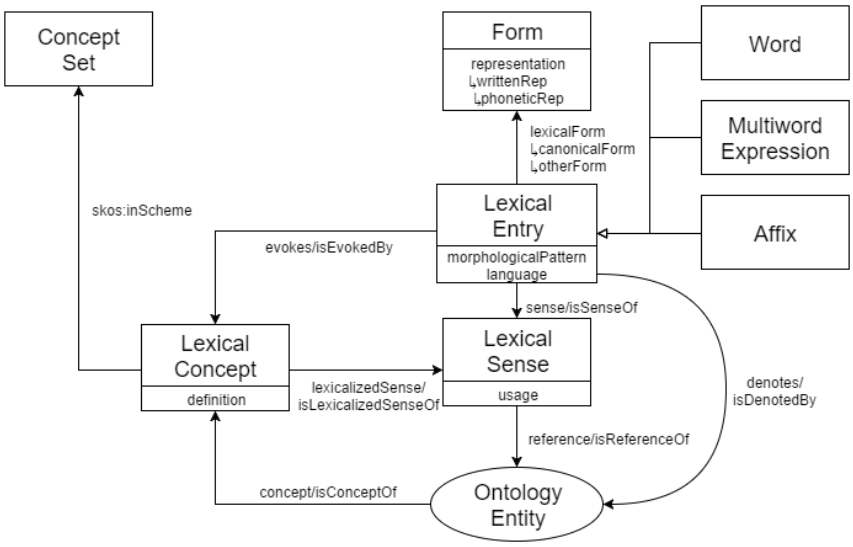

Figure 1: OntoLex-Lemon, the OntoLex core vocabulary, https://www.w3.org/2016/05/ontolex/#core

resources), recent development of LLOD has been rapid, and especially so in the field of lexical resources. To a large extent, this has been enabled by synergies between distributed efforts to data publication and linking that were possible due to the wide reception of a common community standard for lexical data on the web of data, the OntoLex vocabulary.

2.1. OntoLex

OntoLex [1] is a widely used community standard for machine-readable lexical resources, and the dominant vocabulary for modeling machine-readable dictionaries as Linked Data. The goal of OntoLex is to represent lexical resources as a knowledge graph, enabling the integration of information from different dictionaries and to facilitate exchange, storage, and reusability of lexical information.

The OntoLex core model (OntoLex-Lemon, Fig. 1) is structured around the concept of lexical entry, which represents a single lexeme and its grammatical properties, inflected forms and word senses. The meanings of these senses can be defined formally by reference to an ontology or informally by a lexical concept. For lexicography, OntoLex provides a framework for data modelling, making dictionaries available as LLOD. This allows to integrate lexical data from different sources, to provide machine-readable versions of classical dictionaries, and linguistically enriched labels for concepts in ontologies or knowledge graphs, or to link lexical information with multimodal content, corpora or other forms of external data. For Natural Language Processing (NLP), OntoLex serves as a semantic layer to link corpus data with lexical resources, and thus to enhance tasks such as text analysis, information retrieval, and machine translation.

For this wide range of applications, OntoLex implements a modular structure, featuring specialized modules for different aspects or use cases of lexical data: **OntoLex-Lime** for lexical metadata (Fiorelli et al., 2015), **OntoLex-SynSem** for syntactic frames and semantic relations (Villegas and Bel, 2015), **OntoLex-VarTrans** for translation and similar relations (Bosque-Gil et al., 2015), **OntoLex-Decomp** for the composition of lexical entries (McCrae et al., 2016), **OntoLex-Lexicog** for the structure of lexical resources (McCrae et al., 2017), and the emerging **OntoLex-Morph** for aspects of linguistic morphology (Chiarcos et al., 2022c). With these, OntoLex provides a standardized and comprehensive framework for representing lexical resources in a machine-readable and interoperable manner, promoting the integration and utilization of lexical information in various applications such as digital lexicography, NLP, and artificial intelligence (Cimiano et al., 2020). In particular, OntoLex can not only formalize lexical resources as static (but queriable) data, but also support the development of web services to access, process, consume or produce such data.

2.2. The FrAC Module

OntoLex-FrAC, or, briefly, FrAC, is another emerging module of OntoLex that enriches lexicons with corpus information (Chiarcos et al., 2022a).[2] FrAC focuses on complementing lexical resources with

[1] https://www.w3.org/2016/05/ontolex

[2] The current draft version of the FrAC specification is found under https://github.com/ontolex/frequency-attestation-corpus-information/.

information derived from corpora, e.g. attestations and frequency, embeddings and distributional similarity, and collocation analysis.

At its top level, OntoLex-FrAC defines two types of objects: **Observables** represent content elements that can be observed or measured in linguistic data, e.g., a lexical entry, a sense, or any other linguistic or conceptual entity that can be identified by a URI and that can be observed in a corpus (given the necessary annotations). **Observations** represent a specific instance of observing or measuring an observable. It captures information about the occurrence or prevalence of the observable in a particular context or dataset. An observation should provide a value (`rdf:value`), and a link to the data source where this observation has been made (`frac:observedIn`). This data source can be any data that provides the context for making observations. This can be any URI that is defined in terms of the DCMI Type Vocabulary,[3] i.e., as collection, dataset, text, image, moving image, physical object, etc.

Depending on the type of observation, OntoLex-FrAC provides designated properties to link observables and observations, e.g., `frac:frequency`, `frac:attestation` and `frac:embedding`:

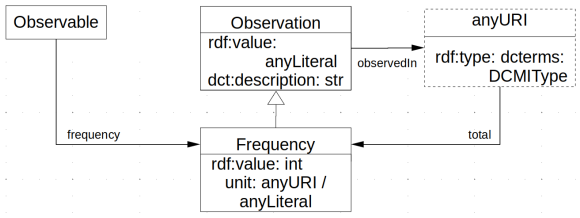

Figure 2: Frequency in OntoLex-FrAC

The **FrAC frequency vocabulary** (Fig. 2) allows to represent frequency counts in a structured manner, as a `frac:frequency` property pointing from an observable to a `frac:Frequency` object. Frequency is an observation whose `rdf:value` is the number of attestations (for the given `frac:unit` of segmentation or annotation, e.g., tokens, sentences, named entities etc.) `frac:observedIn` a particular corpus.

The **FrAC attestation vocabulary** (Fig. 3) characterizes real-world examples in dictionaries, including context snippets, citations, and other information related to the attested evidence. For lexicographic web services, modelling attestations in OntoLex allows to support lexicographers with examples drawn from corpora or literature. Attestations provide a literal excerpt of the underlying data as `rdf:value` (or, if normalized, as `frac:gloss`).

[3]See https://www.dublincore.org/specifications/dublin-core/dcmi-terms/.

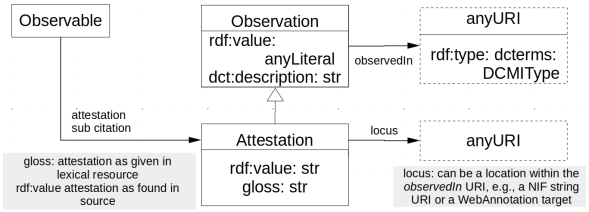

Figure 3: Attestations in OntoLex-FrAC

If an attestation can be linked to a particular element or position within a data source, the property `frac:locus` should provide an unambiguous reference, e.g., a string URI for plain text (Hellmann et al., 2013), an token URI for annotated corpora (Chiarcos and Fäth, 2017), or a Web Annotation selector object for textual and multimodal content on the web (Sanderson et al., 2017).

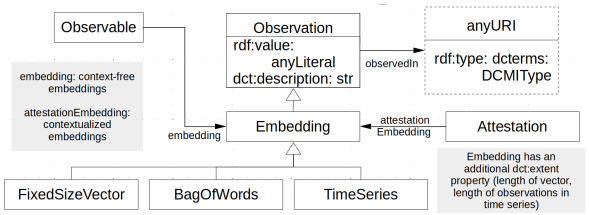

Figure 4: Embeddings in OntoLex-FrAC

The **FrAC embedding vocabulary** (Fig. 4) formalizes the mapping of lexical, semantic or conceptual entities into numerical feature spaces, e.g., using fixed-size vectors, weighted bags of words, or sequences of fixed-size vectors (time series data). With `frac:embedding`, an observable is assigned a static embedding (e.g., a word embedding, sense embedding, or concept embedding). Contextualized embeddings are modelled as the `frac:attestationEmbedding` of a particular attestation of an observable, e.g., a string at a particular `frac:locus` in a given corpus.

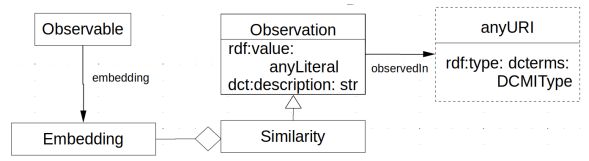

Figure 5: Distributional similarity in OntoLex-FrAC

The **FrAC similarity vocabulary** (Fig. 5) is concerned with distributional similarity of observables, as used in digital lexicography and language technology, e.g., its cosine similarity. As this is calculated from numerical representations of the underlying lexical, semantic or conceptual entities, it is not linked with observables, but with their embeddings (or, bags-of-words or time series). Similarity relations are represented as containers with exactly

two observables (resp., their embeddings), with the `rdf:value` representing the respective similarity metric. Similarity clusters are similarly represented as containers with two or more embeddings.

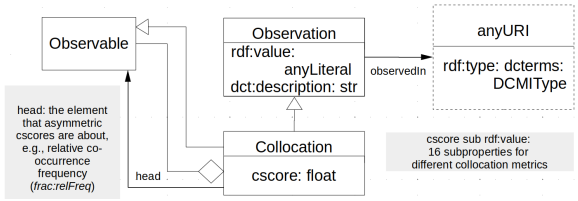

Figure 6: Collocations in OntoLex-FrAC

The **FrAC collocation vocabulary** (Fig. 6) uses a similar container structure, but here, observables are grouped together based on their co-occurrence in a corpus: A FrAC collocation is any pair (or sequence, or set) of expressions (observables) that can be characterized by a collocation score. In practical applications, multiple collocation scores are provided for collocation candidates, and FrAC provides an inventory of standard scores as sub-properties of `frac:cscore`. As it comes with a collocation score observed in a particular corpus, a `frac:Collocation` is an observation. As it can be assigned attestations, corpus frequency, distributional similarity, or be part of larger collocations (e.g., 3-grams), collocations are also within the domains of all these properties, and thus also modelled as observables in FrAC.

The modelling of collocations by Chiarcos et al. (2022b) has been the latest addition to OntoLex-FrAC, so far, and public discussions about its scope and limitations represent one of the motivations for this paper. FrAC collocations are meant to be objects that a lexicographer (by means of an appropriate interface or a designated tool) retrieves from automated corpus analysis, e.g., using a web service comparable to those of *Sketch Engine* (Kilgarriff et al., 2004, 2014) (Lexicom, 2023), which represents the industry standard in the realm of corpus-based lexicography, and plays a pivotal role in corpus linguistics. Accordingly, its collocation scores are mirrorred in sub-properties of `frac:cscore`. Yet, some limitations of the existing vocabulary could not be overcome in this way:

morphosyntactic filters Some collocation metrics measure the collocation probability of two expressions *in a particular grammatical context*. This includes collocation metrics that operate on a reduced set of observables filtered for certain parts of speech, e.g., by excluding stop words and closed class expressions.

syntactic filters Lexicographers often use word sketches (Kilgarriff et al., 2010), i.e., collocations under the condition that a particular grammatical role holds between collocates. In Sketch Engine, this functionality is provided by querying dependency syntax.

distance and context Currently, FrAC allows to specify whether a collocation is ordered (`rdfs:Seq`) or not (`rdfs:Bag`), but there is no formal means to express whether these collocates are meant to be adjacent or just have to appear in the same context window, or what the context window for collocations is.

colligations Certain *syntactic patterns* exhibit similar selection preferences as collocations, albeit not on the level of words, lexemes or even concepts, but on the level of grammatical categories (Firth, 1957). Corpus linguistics thus tends to combine search, retrieval and research of collocation with syntactic patterns and grammatical selection preferences (Sinclair, 1991, p.102-108).

These aspects can be addressed by incorporating corpus queries into FrAC as described in the following section.

3. Modelling Corpus Queries in OntoLex-FrAC

In this paper, we propose an extension of FrAC to also accommodate corpus queries. This follows five main objectives:

1. Provide a community standard, not a tool: Lexicography is still based on domain-specific software solutions, there are several tools and web services that provide this functionality, and many (not all) of them developed as open source projects. Our goal is to make them interoperable and interchangeable to reduce software dependencies. In the context of web technologies, the most widely used RDF vocabulary for digital lexicography is OntoLex-Lemon.

2. Support web services and dynamically generated responses: Digital-born lexical resources are grounded in corpora, and occasionally, they are actually dynamically populated from the underlying corpus. OntoLex-FrAC should be applicable to web services that perform that population task.

3. Support filters and patterns: Existing tools used in lexicographic research and for the creation of dictionaries in the industry often incorporate filters for retrieving frequency, corpus examples (attestations), for the calculation of distributional similarity or collocation scores. When frequencies, attestations, similarity or

collocations are to be returned by an OntoLex-compliant web services, it should be possible to specify these requirements in an interoperable way.

4. Support corpus querying: One way of implementing the aforementioned filters is by means of corpus queries, and, in fact, many tools and APIs used for the purpose support the usage of conventional corpus query languages as an advanced mode of search.

5. Keep it Simple and Generic: The OntoLex community is not the place to standardize corpus querying, instead, it has a focus on dictionaries. So, aim to be unrestricted regarding the query language used, but focus on integrating queries and their results with lexical resources and lexicographic workflows.

In order to meet these requirements, we propose the vocabulary illustrated in Fig. 7. The first and second requirements are motivations for using OntoLex and RDF technologies in the first place, rather than additional constraints imposed on an OntoLex vocabulary for corpus queries. Then, we would argue that both filters and patterns can be encoded by corpus query languages, so that the third objective is actually a sub-case of the fourth. To formalize corpus queries in line with the fifth objective, we refrain from providing explicit semantics for corpus queries, but instead introduce an OntoLex object that can be used to wrap *any* corpus query language expression.

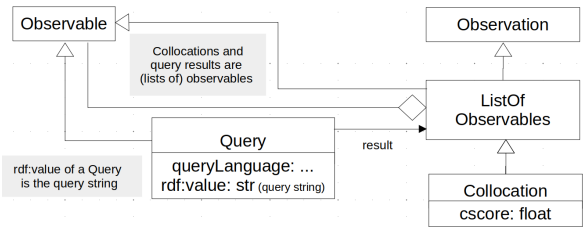

Figure 7: Proposed query extension

We introduce `frac:Query` as a subclass of `frac:Observable`, as corpus queries formalize entities and relations that we *expect to* observe in a corpus. We would then assume that any corpus query can be expressed in a string, provided as the `rdf:value` of the query. In order to ensure that this query can be adequately interpreted, we also need to specify which query language to apply. To this end, we require the property `frac:queryLanguage` to provide a reference to the respective manual, a website with instructions on querying or another unambiguous identifier for a corpus query language as a URI. For the example of querying in SketchEngine, we point, for example, to the documentation of (their specific dialect of) the Corpus Query Language (Christ et al., 1994, CQL): `<https://www.sketchengine.eu/documentation/corpus-querying/>`.

In the context of a web service, a query formalizes the request of a client to a web service. In addition, it can also be used to formalize the response. Then, the query object is returned (with all the information of the request), but complemented with zero or more `frac:result` properties. We would expect that the result of any corpus query can be represented as a table, where variables (or, if no variables are defined, the complete query) are associated with the respective columns. In the existing FrAC vocabulary, collocations have a similar structure, as they represent a list (`rdfs:Seq`) or set (`rdfs:Bag`) of observables. We thus suggest "list of observables" as a generalization over both query results and collocations. For corpus queries, we would assume that every variable (as the query itself) represents an observable. In RDF, generic variables can thus be defined as blank nodes (underspecified values) of `rdf:type frac:Observable`. Albeit it might be underspecified as to what kind of entities it can represent, we would expect it to represent a word (`ontolex:Form`), lemma/lexeme (`ontolex:LexicalEntry`, `lexicog:Entry`), word sense (`ontolex:LexicalSense`), concept (`ontolex:LexicalConcept`), syntactic patterns (`synsem:SyntacticFrame`, `synsem:SyntacticArgument`), translation relation (`vartrans:Translation`, `vartrans:translatableAs`), morphological unit (`morph:Morph`) or any other annotated element in the data (any URI) that OntoLex information can be provided about.

Conceptually, this is very similar to the existing definition of collocations in OntoLex-FrAC, which are just (observable) lists of observables in a particular data source, with the caveat that collocations can be unordered, but that we require a fixed order here, in order to preserve the positional correspondence between observables and the variables in the query string. Assuming that we can limit `frac:Collocation` to list (rather than set) structures (and then, to use the `rdf:value` of an associated `frac:Query` to make the internal structure of the collocation explicit), we propose the novel class `frac:ListOfOfObservables` as a generalization over `frac:Collocation` and define these lists of observables as objects of one or multiple `frac:result` properties that a `frac:Query` object can take in a response.

In a list of observables, the order is aligned with the order or variables in the query expression, and every list of observables is defined to be `frac:observedIn` one particular data source. Like collocations, lists of observables are thus ob-

servations, and their `rdf:value` can be used to provide a confidence score, e.g., match probability or a status code (akin to, say, HTTP code 404) concerning a corpus query, or – for actual collocations –, their collocation score.

For queries against several corpora, multiple `frac:result` objects should be created. Like collocations, a specific list of observables is also defined as an observable, and for the same reasons, i.e., we would like to provide frequency, similarity, or attestation information about them. The actual corpus matches for the complete query are then given in a series of `frac:attestation` properties of the `frac:ListOfObservables` object. The corpus matches of individual variables within a query are then given as `frac:attestation` properties of the `rdfs:member` elements of the `frac:ListOfObservavles`, and by means of their `frac:locus` URIs, matches for individual variables and the overall queries can be easily aligned with each other. As for aggregate queries on the frequency of matches in a corpus, this can be provided analoguously by providing a `frac:frequency` property for a `frac:ListOfObservables`. If the backend provides the means to compute aggregate embeddings for query responses, these can be returned as `frac:embedding` objects, and, again, contextualized embeddings for individual occurrences of an observable (or a number of co-occurring expressions as stated in the query) can be provided via the `frac:attestationEmbedding` properties of the associated attestations.

Finally, the proposed classes `frac:Query` and `frac:ListOfObservables` support the study of and search for colligations and syntactic selection preferences in collocation studies. A colligation proper is then merely defined as a `frac:ListOfObservables` with an internal structure as determined by the `rdf:value` of the associated `frac:Query`.

It should be noted that all of this is achieved by introducing only two additional classes (`frac:Query` and `frac:ListOfObservables`) and two properties (`frac:result`, `frac:queryLanguage`) to OntoLex-FrAC, and all our changes (with the exception of limiting `frac:Collocation` to `rdfs:Seq`) are downward-compatible. Our proposal is thus as minimalistic as possible and yet enables a broad band-width of real-world applications in the context of retrieving corpus information and adding it dynamically to lexical resources, or providing web services for this purpose.

In order to assess its practicality, we also provide working examples for using the FrAC query extension in a real-world use case.

4. Use Case Description

The use cases described below originate from a project TESLA (Text Embeddings - Serbian Language Applications) that involves the development of state-of-the-art language applications for Serbian. Aside from developing NLP resources and tools for the Serbian language, this also includes the development of a web portal, with user-friendly explanatory visualizations, such as language patterns and phenomena detected in texts, aiming to support applications in corpus-based lexicography. Furthermore, the project aims to provide tools for named entity recognition, entity linking, relation extraction, and knowledge graph building. For both corpus-based lexicography and uses of knowledge graphs in NLP, the use of OntoLex and Linguistic Linked Open Data technologies is a well-established best practice to ensure the creation of re-usable and FAIR language resources, it has thus been a design decision made early on in the project. At the same time, the OntoLex-FrAC vocabulary development is expected to conclude in early 2024, only, thus allowing us to propose and test the query extension of OntoLex-FrAC in this paper, in due time to still be considered to be discussed by the OntoLex community and added to the community report in preparation.

We base our work on established technologies, most notably, Sketch Engine. As described above, Sketch Engine is a widely used tool that is commonly accessed by researchers and linguists to explore and analyze linguistic data contained within large corpora. The Sketch Engine API access offers a convenient means of harnessing some of Sketch Engine's functionality for linguistic investigations, making it a versatile resource for various scientific inquiries, but the access is on a commercial basis. In addition to the proprietary Sketch Engine, its developers thus also provide the *NoSketch Engine* (Lexicom, 2022) as an open-source counterpart, albeit with a limited set of functionalities.

With respect to accessing corpus data, Sketch Engine and NoSketch Engine provide similar functionalities, but they differ in their level of support for users and clients. Sketch Engine is usually used in public installation and with the data hosted by its developers, whereas NoSketch Engine is an open-source counterpart with a limited set of functionalities and without preloaded corpora. Researchers can use it with their own corpora to make request calls to interact with the NoSketch system enabling incorporation of corpus queries into their computational workflows, allowing for seamless data extraction and analysis as part of a more extensive infrastructure.

4.1. Wrapping Sketch Engine Responses

For query demonstrations in *Sketch Engine*, the British National Corpus (BNC) (BNC, 2023) [4] is used, especially because it has a permissive license and its use provides reproducibility. The BNC is a 100-million-word collection of samples of the written and spoken language of British English from the later part of the 20th century (BNC, 2023).[5]

```
:BNC a owl:Class ;
  rdfs:subClassOf [ a owl:Restriction ;
  owl:hasValue <https://
    app.sketchengine.eu/#dashboard
    ?corpname=preloaded/bnc2_tt31>;
  owl:onProperty frac:corpus ] .
```

The following listing presents an example of a query `:q_coll_risk_n` that extracts collocation candidates for noun *risk* from `:BNC`. The bare corpus query, provided as `rdf:value` of the query, uses the CQL query language, but it does not provide the actual access point. For this purpose, we define that our query object should be the `owl:sameAs` the URI that performs the HTTP GET request against the Sketch Engine end point.[6] The endpoint with *base_url*= `https://api.sketchengine.eu/bonito/run.cgi` and service *collx* requires the user authentication, implemented with *USERNAME* and *API_KEY*.

```
:q_coll_risk_n a frac:Query ;
 rdf:value '[lemma="risk"&tag="N.*"]' ;
 owl:sameAs
 <https://api.sketchengine.eu/
   bonito/run.cgi/collx
   ?q=q[lemma="risk"&tag="N.*"]
   &corpname=preloaded/bnc2">;
 frac:corpus :BNC ;
 frac:queryLanguage
  <https://www.sketchengine.eu/documentat
  ion/corpus-querying/> ;
 frac:result :r_coll_risk_n .
```

As this example shows, FrAC provides the structural anchors for incorporating this query into an RDF dataset or a service, as well as the necessary metadata to run and interpret the query string.

In the response, a web service can now return the same query but complemented with an additional `frac:result` property, which mirrors the original response of the Sketch Engine API as given below for the collocation candidate *risk+reduce*:

```
{'str': 'reduce',
 'freq': 249,
 'coll_freq': 6941,
  'Stats': [{'s': '15.72959', 'n': 't'},
   {'s': '8.29771', 'n': 'm'},
   {'s': '8.69190', 'n': 'd'}],
'pfilter':'q=P-5+5+1+[word="reduce"]',
'nfilter':'q=N-5+5+1+[word="reduce"]'},}
```

In the response, the collocation candidate *reduce* is given, its absolute frequency in the default context window (`freq`, as defined by `pfilter` and `nfilter`), the total number of cooccurrences of the collocate in the corpus (`coll_freq`) and different collocation scores that correspond to `frac:tScore`, `frac:pmi` and `frac:logDice`, respectively, (Kilgarriff et al., 2014) which already are established subproperties of `frac:cscore`.

The corresponding objects of `frac:result`, automatically construed by a wrapper around the Sketch Engine API response, are provided as a series of `frac:ListOfObservables`s of two variables, corresponding to elements matching the (lexical entry) `risk` and (the tag) `N.*`:[7]

```
:r_coll_le_risk_n_1 a rdf:Seq,
    frac:ListOfObservables ;
    rdf:_1 :coll_le_risk ;
    rdf:_2 :coll_le_reduce;
    frac:obsevedIn :BNC.
:r_coll_le_risk_n_2 a rdf:Seq,
    frac:ListOfObservables ;
    rdf:_1 :coll_le_risk ;
    rdf:_2 :coll_le_factors;
    frac:obsevedIn :BNC. ...
```

The Sketch Engine endpoint `/collx` used here returns collocation scores, so, these co-occurrences actually are collocations in terms of FrAC. By providing the collocation scores, we can leave this implicit, though. For *risk+reduce*, the following triples encode collocation scores:

```
:r_coll_le_risk_n_1
  frac:head :coll_le_risk;
  frac:tScore "15.72959";
  frac:pmi "8.29771";
  frac:logDice "8.69190".
```

[4] https://app.sketchengine.eu/#ca?corpname=preloaded%2Fbnc2_tt31

[5] Code for replicating this use-case is available at https://github.com/ontolex/frequency-attestation-corpus-information/blob/master/samples/queries/sketch-engine-api/sketch-api-frac-no-API-KEY.ipynb

[6] Alternatively, a user can use the retrieval URI directly as the URI for the query, but separating both allows us to provide resolvable query URIs for query end points that do not directly provide an RDF response.

[7] Note that a list of observables can group together any number of attestations of the respective observables, in this case, lexical entries. But developers can decide to not return individual lexical entries, but, for example, just treat each variable in a query as an observable in its own right, the list of observables would just be the list of variables, so there would be one `frac:listOfObservables` per query per data source. We went for a more informative structuring of the result, here, but the query extension is designed to account for both approaches.

Two different frequencies are returned, each requiring a separate frequency object:

```
:r_coll_le_risk_n_1 frac:frequency
  [ dct:description "coll_freq";
     rdf:value "6941" ],
  [ dct:description "freq, pfilter:...";
     rdf:value"249" ] .
```

With RDFS semantics, the type of `frac:frequency` objects can be left implicit and is inferred to be `frac:Frequency`. In a more concise, and more explicit representation, designated subclasses of frequency can be created. Note that our rendering of filter conditions is not machine-readable. If this is to be achieved, an explicit corpus query can be used.

Other public Sketch Engine end points[8] can be addressed (and wrapped) analogously. The retrieval of corpus examples is illustrated using NoSketch Engine below.

4.2. Queries with NoSketch Engine

Collocation queries were illustrated against public Sketch Engine data to facilitate reproducibility by reviewers. On our own data, we illustrate full-fledged corpus queries for the word form *fudbal* in the corpus `:SrFudKo` (labeled `SK21` in examples) installed on https://noske.jerteh.rs/. As before, the property *rdf:value* points to a CQL string designed to return complete sentences that contain noun word *fudbal* (football): *"<s/> containing [lc="fudbal" & tag="N.*"]"*. The `frac:result` included in the response is a `frac:ListOfObservables` as before, but here, it carries one `frac:attestation` property per corpus match.

```
:q_att_SK21_fudbal4 a frac:Query;
  rdf:value "<s/> containing
  [lc="fudbal" & tag="N.*"]";
  owl:sameAs <https://noske.jerteh.rs/
   run.cgi/first?corpname=SK21&tab=...>;
  frac:queryLanguage
  <https://www.sketchengine.eu/
    documentation/corpus-querying/>;
  frac:result [a frac:ListOfObservables ;
    frac:attestation :att_1_SK21_fudbal4;
    frac:attestation :att_2_SK21_fudbal4;
    frac:attestation :att_3_SK21_fudbal4;
     ...] .
```

An examplary `frac:Attestation` from the results of the previous query is given in the following listing. The sentence: *Znači, da čekanje brže prođe, fudbal uvek dobro dođe!* means "So, to make the wait go by faster, football always comes in handy!". The property *frac:locus* returns the string URI specifying the portion of the corpus from which the sentence was retrieved.

[8] https://www.sketchengine.eu/apidoc/#/

```
:att_1_SK21_fudbal4 a frac:Attestation;
  rdf:value "Znači, da čekanje brže
  prođe, fudbal uvek dobro dođe!";
  frac:observedIn <https://noske.
  jerteh.rs/#dashboard?corpname=SK21>;
  frac:locus <http://...>
```

The following query for frequency and its response object are analogous, except for, using the `/freq` end point of our NoSketch instance:

```
:q_fr_SK21_fudbal4 a frac:Query;
 rdf:value "[lc='fudbal' & tag='N.*']";
 owl:sameAs <https://noske.jerteh.rs/
   run.cgi/freqs?...>;
 frac:queryLanguage
  <https://www.sketchengine.eu/
  documentation/corpus-querying/>;
 frac:result [
    a frac:ListOfObservables ;
    frac:frequency :freq_SK21_fudbal1].

:freq_SK21_fudbal1 a frac:Frequency;
    rdf:value 19016 ;
    frac:observedIn <https://noske.
    jerteh.rs/#dashboard?corpname=SK21>
```

A key difference between both queries is that they run against different endpoints of the NoSketch installation, encoded in the `owl:sameAs` URIs.

This section described how corpus queries with (No)Sketch Engine can be wrapped in FrAC for collocation analysis, retrieving corpus matches, and frequency counts. For features not supported by Sketch Engine, we provided analogous queries against our NoSketch Engine installation with URI placeholders. At the moment, these functionalities are implemented by the web services designed for communication between the platform that is to be developed in the course of the project and our NoSketch Engine instance. Frontend development is subject to on-going implementation efforts. A key benefit we expect from using OntoLex-FrAC for the information exchange between the frontend and backend is that the backend components become easily replaceable with any corpus management system that supports CQL language.

5. Outlook and Summary

OntoLex is a relatively widely used vocabulary for lexical resources, but mostly appreciated in contexts in which links between distributed datasets are to provided and/or exploited. This focus on modelling *data* and on information integration across different datasets sets it apart from earlier, primarily XML-based approaches that focus on modelling *dictionaries* as stand-alone objects, and although it creates some overhead, this also opens new possibilities. One such possibility is to link lexical resources with dynamic web services or external resources such as corpora, and to do so in a

W3C-standardized, and thus portable and widely supported way.

We provide minimalistic modeling of corpus queries in the context of OntoLex-FrAC by means of four novel vocabulary elements: **frac:Query** is the query object, with the query string as its `rdf:value`; **frac:queryLanguage** providing an URI that identifies the query language of the query string, and thus helps the backend to confirm whether this query can be executed and how to execute it; **frac:result** points to the results of a corpus query as observed in one particular corpus, and **frac:ListOfObservables**, the object type of corpus query results, formalizing a table where the query itself and variables (anonymous observables) represent columns, individual `frac:attestation` properties represent corpus matches, and `frac:frequency` properties represent aggregate counts. Beyond that, data structures and semantics of corpus queries and their results can be expressed with the vocabulary already established in OntoLex-FrAC. In that regard, our proposal is minimalistic and downward-compatible, and yet addresses a novel use case.

A valid question in this regard is why an RDF vocabulary is needed for that purpose, though, as other formalisms commonly used for web service APIs, say, JSON, could account for corpus querying, as well. We do not debate this idea, but, to some extent, it is based on a misconception: JSON is a specific data format, whereas RDF is a generic data model that can be serialized in many ways. These serializations include a serialization of RDF data in JSON: JSON-LD is an RDF vocabulary that is based on the use of `context` elements that can be added to any JSON dict and that provide a formal RDF interpretation of JSON terms occurring in that dict.

There is a minor limitation regarding OntoLex-FrAC compatibility, though: we propose to redefine `frac:Collocation` to be a subclass of `frac:ListOfObservables` in order to limit it to *ordered* lists of observables (previously they could also be unordered sets). This is necessary to encode the tabular structure of query results, where one match is provided after the other, but using the novel query object and the expressive power of established corpus queries, we can now make the exact sequential and grammatical structure of collocations explicit with the `rdf:value` of an associated query object. This is because we can use the query language to encode the exact grammatical relations between the collocates ... within the limits of the respective query language.[9]

[9]With CQP, as used here, we would be limited to word-level annotations and positional information. More advanced corpus query languages, e.g., ANNIS QL (Chiarcos et al., 2008) or TGrep (Ghodke and Bird, 2010), would allow us include syntactic dependencies or even more complicated structures.

At the moment, this association is encoded explicitly only if a query object is provided along with a collocation, but beyond the context of corpus query inputs, would like to propose yet another property to be added to OntoLex-FrAC: a property like `frac:pattern` could be used to link a collocation with a `frac:Query` if a collocation object was not created in response to a query. However, this addition is beyond the scope of this paper and can be further discussed with the OntoLex community and the participating lexicographers.

As for potential uses of the vocabulary, we foresee applications in the development of portals, web services, and clients for lexicographic research, and thus a contribution to any downstream application of corpus-based lexicography, whether connected to language technology, language learning, or research. Furthermore, the same specifications can be used to address the linking between corpus resources and lexical resources in tools and platforms for corpus linguistics.

At the moment, digital lexicography is based on domain-specific software, specifically designed to facilitate the retrieval of lexicographically relevant information from mid- to large-scale corpora. We described word sketches as a well-known concept in this context. NoSketch Engine does not provide this functionality, but it can be replicated with the combination of appropriate CQL queries and some post-processing – and the queries can be represented using the vocabulary proposed in this paper, which provides a layer of interoperability between different corpus managers. While CQL is widely supported among corpus management systems (Christ et al., 1994; Dura, 2006; Hardie, 2012; Machálek, 2020; Ionov et al., 2020), this does not even depend on it, since any other corpus query language could be used in place of it – as long as it can be represented in a query string and its documentation can be referred to with a URI. In combination with efforts to create an abstraction layer for corpus query languages like CQLF (Evert et al., 2020), this can lead to true interoperability.

6. Acknowledgements

This research was supported by the COST Action NexusLinguarum (CA18209) – “European network for Webcentered linguistic data science” and by the Science Fund of the Republic of Serbia, #7276, Text Embeddings - Serbian Language Applications - TESLA. We would like to thank the OntoLex W3C community and the participants of OntoLex-FrAC modelling discussions and three anonymous reviewers for valuable input and feedback.

7. Bibliographical References

Christian Bizer, Tom Heath, and Tim Berners-Lee. 2023. Linked Data – The story so far. In *Linking the World's Information: Essays on Tim Berners-Lee's Invention of the World Wide Web*, pages 115–143. Association for Computing Machinery, New York, NY, United States.

Julia Bosque-Gil, Jorge Gracia, Guadalupe Aguado-de Cea, and Elena Montiel-Ponsoda. 2015. Applying the OntoLex model to a multilingual terminological resource. In *The Semantic Web: ESWC 2015 Satellite Events: ESWC 2015 Satellite Events, Portorož, Slovenia, May 31–June 4, 2015, Revised Selected Papers 12*, pages 283–294. Springer.

Christian Chiarcos, Elena-Simona Apostol, Besim Kabashi, and Ciprian-Octavian Truică. 2022a. Modelling frequency, attestation, and corpus-based information with OntoLex-FrAC. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4018–4027.

Christian Chiarcos, Stefanie Dipper, Michael Götze, Ulf Leser, Anke Lüdeling, Julia Ritz, and Manfred Stede. 2008. A flexible framework for integrating annotations from different tools and tag sets. In *Traitement Automatique des Langues, Volume 49, Numéro 2: Plate-formes pour le traitement automatique des langues [Platforms for Natural Language Processing]*, pages 217–246.

Christian Chiarcos and Christian Fäth. 2017. CoNLL-RDF: Linked corpora done in an NLP-friendly way. In *Language, Data, and Knowledge: First International Conference, LDK 2017, Galway, Ireland, June 19-20, 2017, Proceedings 1*, pages 74–88. Springer.

Christian Chiarcos, Katerina Gkirtzou, Maxim Ionov, Besim Kabashi, Anas Fahad Khan, and Ciprian-Octavian Truica. 2022b. Modelling collocations in OntoLex-FrAC. In *LREC 2022 Workshop Language Resources and Evaluation Conference 20-25 June 2022*, page 10.

Christian Chiarcos, Katerina Gkirtzou, Anas Fahad Khan, Penny Labropoulou, Marco Passarotti, and Matteo Pellegrini. 2022c. Computational morphology with OntoLex-Morph. In *LREC 2022 Workshop Language Resources and Evaluation Conference 20-25 June 2022*, page 78.

Oliver Christ, Bruno M Schulze, Anja Hofmann, and Esther Kõnig. 1994. The IMS Corpus Workbench. *Institut fr Maschinelle Sprachverarbeitung, Universitt Stuttgart*.

Philipp Cimiano, Christian Chiarcos, John P McCrae, and Jorge Gracia. 2020. *Linguistic Linked Data. Representation, Generation and Applications*. Springer.

Elzbieta Dura. 2006. CULLER – A user-friendly corpus query system. In *DWS 2006: Proceedings of the Fourth International Workshop on Dictionary Writing Systems*, page 47.

Stefan Evert, Oleg Harlamov, Philipp Heinrich, and Piotr Bański. 2020. Corpus Query Lingua Franca part ii: Ontology. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 3346–3352.

Manuel Fiorelli, Armando Stellato, John P McCrae, Philipp Cimiano, and Maria Teresa Pazienza. 2015. LIME: The metadata module for OntoLex. In *The Semantic Web. Latest Advances and New Domains: 12th European Semantic Web Conference, ESWC 2015, Portoroz, Slovenia, May 31–June 4, 2015. Proceedings 12*, pages 321–336. Springer.

John R. Firth. 1957. *Papers in Linguistics*. Oxford University Press, Oxford.

Sumukh Ghodke and Steven Bird. 2010. Fast query for large treebanks. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 267–275.

Andrew Hardie. 2012. CQPweb – Combining power, flexibility and usability in a corpus analysis tool. *International journal of corpus linguistics*, 17(3):380–409.

Sebastian Hellmann, Jens Lehmann, Sören Auer, and Martin Brümmer. 2013. Integrating NLP using Linked Data. In *The Semantic Web–ISWC 2013: 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part II 12*, pages 98–113. Springer.

Maxim Ionov, Florian Stein, Sagar Sehgal, and Christian Chiarcos. 2020. CQP4RDF: Towards a suite for RDF-based corpus linguistics. In *The Semantic Web: ESWC 2020 Satellite Events: ESWC 2020 Satellite Events, Heraklion, Crete, Greece, May 31–June 4, 2020, Revised Selected Papers 17*, pages 115–121. Springer.

Adam Kilgarriff, Vít Baisa, Jan Bušta, Miloš Jakubíček, Vojtěch Kovář, Jan Michelfeit, Pavel Rychlý, and Vít Suchomel. 2014. The Sketch Engine: Ten years on. *Lexicography*, 1:7–36.

Adam Kilgarriff, Vojtěch Kovář, Simon Krek, Irena Srdanović, and Carole Tiberius. 2010. A quantitative evaluation of word sketches. *Proceedings*

of the 14th EURALEX International Congress, pages 372–79.

Adam Kilgarriff, Pavel Rychlý, Pavel Smrž, and David Tugwell. 2004. The Sketch Engine. *Proceedings of the 11th EURALEX International Congress*, pages 105–116.

Erik Körner, Thomas Eckart, Axel Herold, Frank Wiegand, Frank Michaelis, Matthias Bremm, Louis Cotgrove, Thorsten Trippel, and Felix Rau. 2023. *Federated Content Search for Lexical Resources (LexFCS): Specification*.

Tomáš Machálek. 2020. KonText: Advanced and flexible corpus query interface. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 7003–7008.

John P McCrae, Julia Bosque-Gil, Jorge Gracia, Paul Buitelaar, and Philipp Cimiano. 2017. The Ontolex-Lemon model: Development and applications. In *Proceedings of eLex 2017 conference*, pages 19–21.

John P McCrae, Philipp Cimiano, Paul Buitelaar, and Georgeta Bordea. 2016. Representing multiword expressions on the web with the OntoLex-Lemon model. In *PARSEME/ENeL workshop on MWE e-lexicons*.

Robert Sanderson, Paolo Ciccarese, and Benjamin Young. 2017. Web Annotation Data Model. Technical report, W3C Recommendation.

John Sinclair. 1991. *Corpus, Concordance, Collocation*. Oxford University Press, Oxford.

Marta Villegas and Núria Bel. 2015. PAROLE/SIMPLE lemon ontology and lexicons. *Semantic Web*, 6(4):363–369.

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, et al. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Scientific data*, 3(1):1–9.

8. Language Resource References

BNC. 2023. *British National Corpus*. British National Corpus Consortium. PID http://www.natcorp.ox.ac.uk.

Lexicom. 2022. *NoSketch Engine*. Lexical Computing CZ s.r.o. Lexicom. PID https://www.sketchengine.eu/nosketch-engine/.

Lexicom. 2023. *Sketch Engine*. Lexical Computing CZ s.r.o. Lexicom. PID https://www.sketchengine.eu.