# A systematic approach to identify the information captured by Knowledge Graph Embeddings

Antonia Ettorre, Anna Bobasheva, Catherine Faron, Franck Michel

# A systematic approach to identify the information captured by Knowledge Graph Embeddings

Antonia Ettorre
aettorre@i3s.unice.fr
Université Côte d'Azur, CNRS, Inria, I3S
Sophia Antipolis, France

Anna Bobasheva
anna.bobasheva@inria.fr
Université Côte d'Azur, CNRS, Inria, I3S
Sophia Antipolis, France

Catherine Faron
faron@i3s.unice.fr
Université Côte d'Azur, CNRS, Inria, I3S
Sophia Antipolis, France

Franck Michel
fmichel@i3s.unice.fr
Université Côte d'Azur, CNRS, Inria, I3S
Sophia Antipolis, France

## ABSTRACT

In the last decade Knowledge Graphs have undergone an impressive expansion, mainly due to their extensive use in AI-related applications, such as query answering or recommender systems. This growth has been powered by the expanding landscape of Graph Embedding techniques, which facilitate the manipulation of the vast and sparse information described by Knowledge Graphs. Graph Embedding algorithms create a low-dimensional vector representation of the elements in the graph, i.e. nodes and edges, suitable as input for Machine Learning tasks. Although their effectiveness has been proved on many occasions and for many contexts, the interpretability of such vector representations remains an open issue. In this work, we aim to tackle this issue by providing a systematic approach to decode and make sense of the knowledge captured by Graph Embeddings. We propose a technique for verifying whether Graph Embeddings are able to encode certain properties of the graph elements and we present a categorization for such properties. We test our approach by evaluating the embeddings computed from the same Knowledge Graph through several embedding techniques. We analyze the results on the level of encoding of each property by all the benchmarked algorithms with the final goal of providing insights into the choice of the most suitable technique for each context and encouraging a more conscious use of such approaches.

## CCS CONCEPTS

• **Information systems → Semantic web description languages**; • **Computing methodologies → Knowledge representation and reasoning**; **Machine learning algorithms**.

## KEYWORDS

Knowledge Graphs, Graph Embeddings, Probing tasks, Explainable AI

## 1 INTRODUCTION

In recent years we have witnessed an impressive growth and expansion in the use of Knowledge Graphs (KGs). More and more projects rely on this kind of representation to store their data without compromising the semantics they bear and, as a result, many large-scale Knowledge Graphs, such as Freebase [6], YAGO [30], Wikidata [32], and DBpedia [3] have been published. The success of this graph representation is mainly due to its usefulness in many AI-related tasks, such as question answering [4, 34], recommendation systems [33], and search [29]; as well as in domain-specific applications, for example in the fields of education [12], medicine [28] and finance [22].

KGs have gained even more visibility since the development of the concept of *Knowledge Graph Embedding (KGE)*, which allows dealing with large-scale KGs more easily. One of the main drawbacks of KGs is the complexity of manipulating the large and sparse information they represent. To tackle this issue, KGE techniques create a fixed-length vector representation of the entities and relations present in the graph. These vectors can then be used for several kinds of tasks within the scope of the KG itself, such as Link prediction [20], Triple Classification [14] and Entity Resolution [21], or in separate downstream applications [9, 13].

One of the main issues related to the usage of Graph Embeddings (GEs) is the difficulty of interpreting them. Indeed, understanding the relationship between the representation in the vector space and the role of the corresponding node (or edge) in the graph is not straightforward. Some work is moving towards more explainable GEs by proposing interpretable embedding methods [18], studying the impact of graph modifications on link prediction [26] or offering explainer approaches for specific embedding models [36]. Nevertheless, a methodology to effectively explain the predictions enabled by KGEs is still missing [24]. Even though several evaluations of the performance of different GE techniques have been conducted with respect to their effectiveness in terms of quality of link prediction [10, 27], to the best of our knowledge no work has been made to systematically analyze the meaningfulness of the information they encode. A first step towards explaining the results of the application of GEs (in any task) lies in understanding the link between the element in the graph (node or edge) and its vector representation. In other words, what is the knowledge present in the graph that can actually be captured and represented through embeddings?

A similar challenge has been addressed in the context of text embeddings, for which the final goal is to understand which characteristics of the language are actually encoded in the embeddings of words and sentences. A first attempt of investigating this research question is based on the use of "probing tasks", presented in [1]. A probing task is an auxiliary classification task that, taking as input the embedding of an element, i.e word, sentence, or node in our case,

is trained to predict the value of a given property for this element. The general idea behind this approach is that, if some information about a given element is encoded through the embedding process in its vector representation, it should be possible to recover such information from the embedding alone.

Inspired by [1], we propose to make use of such auxiliary tasks to verify the hypothesis according to which GEs encode certain properties. We provide a methodology independent from the type, context and final use of the GEs, that allows to systematically analyze the information captured from the KG. We do so by establishing a catalog of probing tasks that can be easily generalized and reused. We focus specifically on RDF knowledge graphs, however the presented approach can be applied to other kinds of graphs, such as property graphs, by defining different probing tasks. We rely on this catalog to evaluate the information encoded by several GE algorithms from a domain-specific KG in e-education describing relations between learning resources. We compare the results achieved by all the algorithms on each task to determine what information is mainly captured by every one of them.

The remainder of this paper is organized as follows. In section 2 we review existing related works. In section 3 we categorize the information that can be encoded by GEs and we propose a corollary organization of probing tasks. In section 4 we compare and discuss the results of state-of-the-art GE algorithms on the identified probing tasks. Finally, conclusions and future works are presented in section 5.

## 2 RELATED WORK

Lately, several efforts have been made to investigate the interpretability of GE techniques. Ying et al. [36] presented GNNExplainer, a tool to provide possible explanations for the results of link prediction through GEs. GNNExplainer is able to identify the subgraph and the node features that are the most relevant for any link prediction made by GNN models. Other approaches towards explainable link prediction have been proposed in [5, 17]. Also, the work proposed in [26] focuses on the interpretability (and robustness) of link prediction by trying to identify the fact whose addition or removal from the KG causes changes in the prediction for a targeted fact. While these works aim for the explanation of link predictions through embeddings, few attempts have been made to understand the content of the embedded representation independently of the downstream application.

Jain et al. [15] raised doubts on the effective capacity of KGEs of capturing the semantics borne by the KG. The authors define a classification and clustering task to predict entity type and compare several well-known KGE algorithms on these tasks, i.e. TransE[7], RESCAL[23], ComplEx[31], DistMult[35] and ConvE[11]. This work concludes that none of these approaches is able to properly encode semantic information, as they can only identify macro classes and not fine-grain ones, e.g. they can distinguish a *person* from a *body_of_water* but not a *scientist* from an *artist*. This approach presents some similarities with the technique used to decode information captured by text embedding introduced by Adi et al. [1], which defines some "probing tasks" to evaluate the information captured by word and sentence embeddings. They provide a set of structural, low-level information properties that can be encoded by the vector representation, i.e. sentence length, word content, and word order. For each one of these properties, they define a classification task that takes as input the word (or sentence) embedding and outputs the value of the property. Through this approach, they evaluate the information encoded by LSTM auto-encoder and CBOW techniques. They also study the changes in the encoded information with the variation of the embeddings' size. In [8], the authors improve the approach proposed by Adi et al. [1], providing a proper classification of 10 probing tasks for the properties possibly encoded by text embeddings, hierarchically organized into surface, syntactic and semantic tasks. Moreover, they redefine the probing tasks to be more general, interpretable, and easily re-usable. To this end, they use as input a single sentence embedding, while Adi et al. [1] used multiple embeddings of both words and sentences. Finally, they benchmark a much wider selection of text embedding algorithms. The categorization defined by Conneau et al. [8] has also been used to uncover the information encoded by BERT, with the specific goal of understanding to which extent this model can capture the structure of the language [16].

Relying on the idea of probing tasks introduced by [1], we present a methodology that allows testing the capacity of KGEs to encode not only entities' types like in [15] but also a vast range of properties and characteristics of graph elements that could possibly be encoded in their vector representations. More precisely, we provide a categorization that hierarchically organizes the various types of knowledge possibly captured by GEs and introduces several probing tasks to assess to which level this knowledge is captured. While the work presented in [5, 17, 26, 36] focuses uniquely on link prediction, the approach we propose can be used to evaluate the meaning of GEs independently of the downstream application. Moreover, we evaluate several GE algorithms, both general-purpose (node2vec) and specifically designed for KGs (TransE, ComplEx, etc.), on this aspect. Additionally, we conduct this evaluation on a real-world KG which allows us to avoid the bias introduced by inverse triples, data redundancy, and cartesian product relationships, normally present in the benchmarking datasets frequently used to evaluate GE algorithms, i.e. FB15k and WN18 [2].

## 3 ANALYSING THE INFORMATION ENCODED BY GRAPH EMBEDDINGS

Following the footsteps of the previously presented works, we introduce a list of auxiliary tasks that could be used to decode the information captured by GEs. An auxiliary task, or probing task, is a simple classification task that takes as input the embedding of an element and outputs the value of a given property for that same element. If it is possible to train a classifier to solve such a task, we can conclude that the set of input features, i.e. the embeddings of the elements, encode the initial property or characteristic. Throughout this process, one would be able to determine whether GEs capture a given characteristic of the elements in the KG, e.g. entities' types or predicates' constraints. Although this is not a *direct translation process* from the embeddings to the set of encoded properties, it would nevertheless represent a first important step towards GEs explainability if we could define a reasonable set of common properties against which to evaluate our GEs.

To do so, we need to identify which kind of information is more commonly contained in a KG and could possibly be captured by the vector representation. Keeping in mind that GE techniques can create a vector for each element in the graph, we need to determine which are the properties of nodes or edges that could be transposed into the vector space. To keep our approach and properties' categories as general as possible and, therefore, adaptable to the large variety of embedding algorithms, our analysis will be limited to the properties of nodes, i.e. *Classes* and *Individuals* in the case of RDF graphs, and we will not take into consideration *Predicates* and *Literals*. Nevertheless, we are aware that the vector representation of a node is computed based on the interactions between such node and all the others. Therefore, we will also introduce some properties that define these interactions and that can be captured by GEs, e.g. presence of a direct link, distance in number of hops, and type of link between two nodes. To evaluate the ability of GE models to catch such information, we will define probing tasks that use as input the embeddings of two nodes, instead of one. We describe hereafter the list of probing tasks, defined on three levels of increasing abstraction, as depicted in Figure 1.

*Structural properties.* With the awareness that a Knowledge Graph is, first of all, a data structure that organizes facts in the form of a graph, we define a first, lowest and most general level of our probing tasks' pyramid: the **structural layer**. We argue that the simplest and most straightforward information that GEs can learn from the graph is its structure. This is a direct consequence of the fact that embedding techniques rely on the presence of direct relations between nodes and, for the approaches based on random walks, on the paths built through those relations. The properties in this category are the ones that every node has, independently of its type or role, and whose definition does not require any additional information other than the list of edges. Examples of properties in this category are *in-degree* (i.e. number of incoming edges) and *out-degree* (number of outgoing edges) for each node. These properties are inherently independent of the context and can be reapplied to any graph.

*Semantic properties.* At a higher level of abstraction, we find properties whose definition makes use of additional information other than the graph structure itself as they require capturing the heterogeneity of the content of the graph. Probing tasks in this class will allow us to test the capacity of GEs to encode **semantic information**, such as *type of nodes and edges*. Here again, these properties are inherently independent of the KG and can be reapplied to any KG. Techniques specifically designed for computing GEs from Knowledge Graphs, such as TransE, ComplEx, DistMult, etc. claim to be able to encapsulate this kind of information.

*Context-specific properties.* This group includes properties that present the highest level of abstraction as they disregard any information about the structure and organization of the data and are uniquely **based on the human understanding of the context**. In other words, those are the characteristics of the elements described in the graph that any individual would intuitively consider relevant for a given task and that would be meaningful for the final use of the embedded representation. For example, let us suppose that we want to rely on GEs to develop a recommender system to suggest restaurants to users based on their tastes. In this case, we would imagine the embeddings of restaurants' nodes to encode the information about the "cuisine" prepared by each restaurant (e.g. Italian, French, etc.), since based on our personal experience this is relevant information when choosing where to eat. While the two previous categories of tasks are generic and easily reusable to analyze GEs over KGs in different contexts, the properties in this last class need to be defined for each specific use-case, based on the knowledge represented in the graph and on the final goal to be achieved through the use of embeddings.

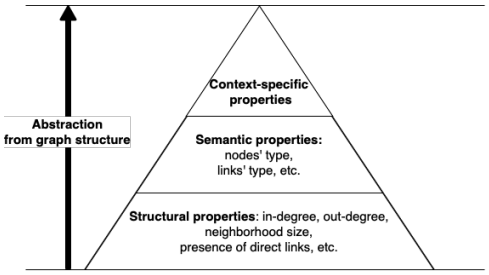


**Figure 1: Categories of probing tasks.**

## 4 EVALUATING GRAPH EMBEDDING ALGORITHMS WITH PROBING TASKS

In this section, we present the results of the application of probing tasks to the OntoSIDES KG [25]. We rely on the presented probing tasks' categories to evaluate the information captured from the graph by several GE models. We compare them and suggest possible explanations for their behavior with the final goal of gathering meaningful insights to help a more informed use of such techniques in different contexts.

### 4.1 OntoSIDES Knowledge Graph

OntoSIDES is a KG that supports the French higher education system for medical studies. In France, medical schools and institutions rely on a common web platform that allows sharing educational content, evaluation resources, students' training traces, and test results. All these data are represented in an RDF graph relying on an OWL ontology. The main knowledge available in the graph is the description of questions, annotated with the related medical specialty and treated subjects, and the description of students including information about their university and their interactions with learning material. Figure 2 depicts the RDF graph representing these elements.

The OntoSIDES graph currently includes the description of 569,762,878 answers to 1,797,180 questions related to 31 medical specialties and given by 173,533 students. In total, the KG contains more than 9.2 billion triples.

In the following experiments, we aim to assess what portion of this knowledge is actually captured and summarized by GEs. Therefore, we rely on the defined probing tasks to analyze the GEs computed on OntoSIDES. More precisely, we evaluate the GEs on the following probing tasks:

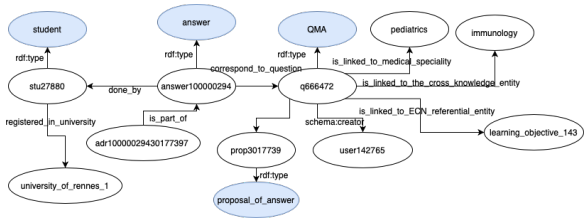


**Figure 2: RDF graph describing the main elements in OntoSIDES. Blue bubbles are `owl:Classes` and white bubbles are instances.**

- **Structural properties:** *in-degree* and *out-degree* for each node and *presence of direct link* between two nodes;
- **Semantic properties:** *type* (`rdf:type`) of each entity and *relation* linking two entities;
- **Context-specific properties:** for each student's node the university the student attends and for each question's node the topic (predicate *learning_objective*) related to that question.

The dimension of the OntoSIDES KG is prohibitive for the computation of GEs. Furthermore, the graph is unevenly distributed w.r.t. the number of answers per students and questions. Therefore, we decided to limit our experiments to a subgraph related to the pediatrics medical specialty. We picked this specialty because it is the one presenting the largest number of attempts by students to answer questions. This choice allows us to obtain a dataset large and heterogeneous enough to train and test all the needed classifiers. Indeed, the size and distribution of the training set are essential to building a high-quality classifier. For example, to be able to associate students to their university, we need a reasonable number of students approximately uniformly distributed among the different universities. Similar considerations hold for all the properties to be probed. Keeping in mind these limitations, we finally extracted a subgraph containing 568.792 entities, and 16 types of relations.

## 4.2 Classification Model

If our hypothesis is valid and the information we aim to retrieve is indeed available in the input features, even a very simple classifier will likely be able to decode it. For this reason, we decided to rely on *Logistic Regression* as a classification model. Hence, we trained a logistic regression model for each one of the probing tasks to be tested.

One of the main issues encountered during the training of such classifiers is the presence of strongly unbalanced classes. As described in Section 4.1, extracting a subset of data that would be balanced w.r.t. all the properties to be tested is not straightforward. Therefore we had to consider, for each property, only the subset of its possible values having a number of occurrences above a threshold. At the same time, we needed to undersample the most frequent classes to avoid a highly skewed distribution.

All the classifiers have been trained with a 5-folds cross-validation on 80% of the data and tested on the remaining 20%.

## 4.3 Graph Embeddings Models

We tested and compared some of the most widely used GE algorithms to analyze the differences in their capacity of capturing information from the graph. We evaluated one of the first GE algorithms conceived for generic graphs, i.e. node2vec; and several algorithms specifically designed for KGs: TransE, ComplEx, DistMult, RESCAL, and RotatE. For the computation of the node2vec embeddings we used the implementation provided by the SNAP framework[1] [19], while for the other KGEs, we relied on the AWS DGL-KGE library[2] [37]. For all the models the dimension of the computed vectors was 100.

## 4.4 Results and discussion

Table 1 reports the results in terms of weighted F1-score obtained for the GEs computed from OntoSIDES on all the probed properties described in Section 4.1. The GE models with the highest F1-score are highlighted in bold.

*Structural properties:* Table 1 shows that no GE model is able to capture the information about the number of incoming and outgoing links for each node. Indeed, all the tested algorithms present comparable but very poor F1-scores, with the best model (node2vec) achieving an F1-score of 0.22 for the prediction of the in-degree and 0.51 for the out-degree. The better results obtained for the out-degree can be explained by the smaller number of classes considered in this case: the maximum possible number of outgoing links in the chosen OntoSIDES subgraph is 13, while the largest in-degree is 35. By taking a closer look at the confusion matrices obtained on these tasks by each algorithm, we notice that all the GE models expose a common behavior for the two prediction tasks: **they are very good in identifying leaves and roots nodes, i.e. nodes with, respectively, in-degree and out-degree equal 0**.

Regarding the prediction of the existence of a direct link between two nodes, Table 1 shows that **all the tested models are fairly good in encoding this kind of information**, with the worst model (RotatE) achieving a F1-score of 0.67 and the best one (node2vec) getting close to 0.9. Being able to train a classifier for such property confirms that embeddings of linked nodes present some common patterns that permit to determine the existence of a connection between the two nodes.

*Semantic properties:* The F1-scores reported in Table 1 reveal the ability of GEs to encode semantic information. These results show that **all the models are capable of identifying the correct type for the majority of nodes in the graph**, with the worst model (RESCAL) achieving a F1-score of 0.87; while **the identification of correct type of relation existing between two nodes is almost perfect for all the models.** This result does not come unexpectedly for models specifically designed for Knowledge Graph Embeddings. In fact, KGE models can partially take into account the semantics of the KG by considering different types of relations and computing a vector representation for every one of them. Surprisingly enough, although it does not rely on any semantic information, node2vec obtained the best results in these tasks: F1 = 0.98 for entity type and F1 = 1.00 for relation type. A possible explanation for this behavior could lie in the fact that entities and relations of the same type possibly present the same connectivity patterns (similar links, common nodes, etc.) that

[1] http://snap.stanford.edu/index.html
[2] https://aws-dglke.readthedocs.io/en/latest/index.html

**Table 1: Weighted F1-scores on the different probing tasks for KGE computed on OntoSIDES.**

| GE algorithm | Structural properties | | | Semantic properties | | Context-specific properties | |
|---|---|---|---|---|---|---|---|
| | **in-degree** | **out-degree** | **direct link** | **entity type** | **relation** | **student's university** | **question's topic** |
| **node2vec** | **0.22** | **0.51** | **0.87** | **0.98** | **1.00** | **1.00** | **0.69** |
| **ComplEx** | 0.11 | 0.37 | 0.74 | 0.90 | 0.99 | 0.51 | 0.18 |
| **TransE** | 0.13 | 0.41 | 0.80 | 0.96 | **1.00** | 0.05 | 0.03 |
| **DistMult** | 0.12 | 0.41 | 0.83 | 0.94 | **1.00** | 0.34 | 0.10 |
| **RESCAL** | 0.13 | 0.37 | 0.83 | 0.87 | 0.99 | 0.08 | 0.06 |
| **RotatE** | 0.11 | 0.37 | 0.67 | 0.95 | **1.00** | 0.57 | 0.37 |

can be captured by the node2vec algorithm and transposed into the vector representation of the corresponding nodes. For example, several nodes representing answers will always be connected to the class *sides:answer* and to a node representing the student that will be linked to the class *sides:student*. Therefore, an instance of an answer can be identified by the direct link to *sides:answer* and the 2-hops path to *sides:student*. At the same time, the space of possible relations existing between nodes will be limited. For example, a *sides:answer* and a *sides:student* might be linked with a single predicate, i.e. *sides:done_by*.

The extremely high values of F1-score obtained for the properties in this category are very likely due to the very low heterogeneity of the knowledge represented in this graph, i.e. the number of different classes and predicates is very limited.

*Context-specific properties.* On the one hand, Table 1 shows that context-specific properties are on average hardly captured by GE methods specifically designed for KGs. While for a human being it is straightforward to understand that the attended university is a relevant concept when summarising information about students, it is much more difficult for the GEs to grasp such a correlation uniquely from the graph. If no additional information (e.g. edge weights) is provided, it is not possible to recognize that, for a student node, the link towards the university is more meaningful than the connection with a given answer. The poor results in terms of F1-score shown by most of the models in Table 1 confirm that **this kind of knowledge presents a much higher level of abstraction (complexity) compared to other previously discussed properties and, therefore, cannot be easily interpreted by such techniques**. Possibly considering longer embeddings could facilitate the incorporation of this kind of knowledge.

On the other hand, we can observe that node2vec achieves good performance also in this task. This could be motivated by the fact that this algorithm strongly relies on the structure of the graph by creating fixed-length paths starting at each node, hence it can recognize the more important role of universities' nodes under a structural point of view. Universities nodes behave as hubs for students, i.e. several students are directly linked to a single university, therefore all the embeddings of students attending the same university will share similar patterns as they are connected in the graph through the university node. The same considerations hold for questions and their corresponding topics, with the difference that encoding topics information is a more challenging task as questions can be linked to several topics and the number of different topics is much larger than the number of universities (more than 500 topics vs. 33 universities). This result highlights the crucial role of the graph structure for the meaningfulness of the final GEs and suggests that careful ontology modeling choices are of the uttermost importance. It is interesting to point out that RotatE and ComplEx perform much better on this task when compared to the other KGE methods. This could be related to their ability to represent entities through Complex numbers, but further experiments are needed to confirm this intuition.

## 5 CONCLUSION AND FUTURE WORK

In this work, we presented a methodology based on probing tasks to verify whether GEs are able to encode certain properties of the graph elements and we introduced a categorization of such properties. To test our approach we proposed a first set of properties that could be encoded by GEs and we relied on them to analyze the knowledge captured by GEs on a real-world Knowledge Graph. We found out that, in our use-cases, GEs can encode information about characteristics of graph elements at any level of abstraction: structural, semantic, and context-specific. We have shown that, even with some differences, all the tested GE models are able to capture the type of a given node, the presence of a direct link, and the type of relationship between two nodes. Moreover, we discovered that all the models are very good at identifying roots and leaves nodes. Finally, we observed that context-specific properties are better captured by node2vec embeddings. We hypothesized that this is because, in many cases, properties that are considered relevant from a human perspective are represented in the graph by nodes with a central structural role. This conclusion highlights the importance of the graph structure and ontology design.

As a future work, our first step will be to extend the presented categorization with additional properties, e.g. `range` and `domain` for any `rdf:Property` or the *Class* or *Individual* nature for any node. We also aim to test more complex models as probing task classifiers to unveil possible non-linear dependencies between the computed embeddings and the characteristics of graph elements. We also want to analyze the possible changes in the encoded knowledge with respect to the model hyper-parameters, e.g. vector dimension, random walks length, etc. Moreover, we will study how modifications to the ontology and, therefore, to the graph structure affect the information borne by GEs. The goal of this research is to discover and provide insights into both ontology modeling choices and selection of the best GE model for the context and the final goal to be achieved through the use of KGEs. In the first stage, we will apply these insights to several tasks related to the OntoSIDES

case study, such as predicting students' outcomes to questions or associating learning objectives to the right educational resources.

## ACKNOWLEDGMENTS

This work is supported by the ANR DUNE project SIDES 3.0 (ANR-16-DUNE-0002-02) and the 3IA Côte d'Azur Investments in the Future project (ANR-19-P3IA-0002).

## REFERENCES

[1] Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. (2016).

[2] Farahnaz Akrami, Mohammed Samiul Saeef, Qingheng Zhang, Wei Hu, and Chengkai Li. 2020. Realistic re-evaluation of knowledge graph completion methods: An experimental study. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1995–2010.

[3] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web*, Karl Aberer, Key-Sun Choi, Natasha Noy, Dean Allemang, Kyung-Il Lee, Lyndon Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 722–735.

[4] Hannah Bast and Elmar Haussmann. 2015. More Accurate Question Answering on Freebase. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management* (Melbourne, Australia) *(CIKM '15)*. Association for Computing Machinery, New York, NY, USA, 1431–1440.

[5] Rajarshi Bhowmik and Gerard de Melo. 2020. Explainable link prediction for emerging entities in knowledge graphs. In *International Semantic Web Conference*. Springer, 39–55.

[6] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A Collaboratively Created Graph Database for Structuring Human Knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data* (Vancouver, Canada) *(SIGMOD '08)*. Association for Computing Machinery, New York, NY, USA, 1247–1250.

[7] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013), 2787–2795.

[8] Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070* (2018).

[9] Amine Dadoun, Raphaël Troncy, Olivier Ratier, and Riccardo Petitti. 2019. Location Embeddings for Next Trip Recommendation. In *Companion Proceedings of The 2019 World Wide Web Conference* (San Francisco, USA) *(WWW '19)*. Association for Computing Machinery, New York, NY, USA, 896–903.

[10] Yuanfei Dai, Shiping Wang, Neal N. Xiong, and Wenzhong Guo. 2020. A Survey on Knowledge Graph Embedding: Approaches, Applications and Benchmarks. *Electronics* 9, 5 (2020). https://doi.org/10.3390/electronics9050750

[11] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[12] Patrick Ernst, Cynthia Meng, Amy Siu, and Gerhard Weikum. 2014. KnowLife: A knowledge graph for health and life sciences. In *2014 IEEE 30th International Conference on Data Engineering*. 1254–1257.

[13] Antonia Ettorre, Oscar Rodríguez Rocha, Catherine Faron Zucker, Franck Michel, and Fabien Gandon. 2020. A Knowledge Graph Enhanced Learner Model to Predict Outcomes to Questions in the Medical Field. In *EKAW 2020 - 22nd International Conference on Knowledge Engineering and Knowledge Management*. Bolzano, Italy.

[14] Jun Feng, Minlie Huang, Mingdong Wang, Mantong Zhou, Yu Hao, and Xiaoyan Zhu. 2016. Knowledge graph embedding by flexible translation. In *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

[15] Nitisha Jain, Jan-Christoph Kalo, Wolf-Tilo Balke, and Ralf Krestel. 2021. Do Embeddings Actually Capture Knowledge Graph Semantics?. In *European Semantic Web Conference*. Springer, 143–159.

[16] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah. 2019. What does BERT learn about the structure of language?. In *ACL 2019-57th Annual Meeting of the Association for Computational Linguistics*.

[17] Bo Kang, Jefrey Lijffijt, and Tijl De Bie. 2019. Explaine: An approach for explaining network embedding-based link predictions. (2019).

[18] Seyed Mehran Kazemi and David Poole. 2018. SimplE embedding for link prediction in knowledge graphs. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*. 4289–4300.

[19] Jure Leskovec and Rok Sosič. 2016. Snap: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)* 8, 1 (2016), 1–20.

[20] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

[21] Ying Lin, Han Wang, Jiangning Chen, Tong Wang, Yue Liu, Heng Ji, Yang Liu, and Premkumar Natarajan. 2021. Personalized Entity Resolution with Dynamic Heterogeneous Knowledge Graph Representations. (2021).

[22] Jue Liu, Zhuocheng Lu, and Wei Du. 2019. Combining enterprise knowledge graph and news sentiment analysis for stock price prediction. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*.

[23] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. *Proceedings of the 28th International Conference on Machine Learning, ICML 2011*, 809–816.

[24] Matteo Palmonari and Pasquale Minervini. 2020. Knowledge graph embeddings and explainable AI. *Knowledge Graphs for Explainable Artificial Intelligence: Foundations, Applications and Challenges* 47 (2020), 49.

[25] Olivier Palombi, Fabrice Jouanot, Nafissetou Nziengam, Behrooz Omidvar-Tehrani, Marie-Christine Rousset, and Adam Sanchez. 2019. OntoSIDES: Ontology-based student progress monitoring on the national evaluation system of French Medical Schools. *Artificial intelligence in medicine* 96 (2019), 59–67.

[26] Pouya Pezeshkpour, Yifan Tian, and Sameer Singh. 2019. Investigating robustness and interpretability of link prediction via adversarial modifications. (2019).

[27] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge Graph Embedding for Link Prediction: A Comparative Analysis. *ACM Trans. Knowl. Discov. Data* 15, 2, Article 14 (Jan. 2021), 49 pages.

[28] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. 2017. Learning a Health Knowledge Graph from Electronic Medical Records. *Scientific Reports* 7 (12 2017).

[29] Charlotte Rudnik, Thibault Ehrhart, Olivier Ferret, Denis Teyssou, Raphaël Troncy, and Xavier Tannier. 2019. Searching News Articles Using an Event Knowledge Graph Leveraged by Wikidata. arXiv:1904.05557 [cs.CL]

[30] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web* (Banff, Alberta, Canada) *(WWW '07)*. Association for Computing Machinery, New York, NY, USA, 697–706. https://doi.org/10.1145/1242572.1242667

[31] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction.

[32] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. *Commun. ACM* 57, 10 (Sept. 2014), 78–85.

[33] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Multi-Task Feature Learning for Knowledge Graph Enhanced Recommendation. *WWW '19: The World Wide Web Conference*, 2000–2010.

[34] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question Answering on Freebase via Relation Extraction and Textual Evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, 2326–2336. https://doi.org/10.18653/v1/P16-1220

[35] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases.

[36] Rex Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. *Advances in neural information processing systems* 32 (2019), 9240.

[37] Da Zheng, Xiang Song, Chao Ma, Zeyuan Tan, Zihao Ye, Jin Dong, Hao Xiong, Zheng Zhang, and George Karypis. 2020. Dgl-ke: Training knowledge graph embeddings at scale. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 739–748.