User-friendly Comparison of Similarity Algorithms on Wikidata

Filip Ilievski, Pedro Szekely, Gleb Satyukov, and Amandeep Singh

Information Sciences Institute, University of Southern California
{ilievski,pszekely,gleb,amandeep}@isi.edu

Abstract. While the similarity between two concept words has been evaluated and studied for decades, much less attention has been devoted to algorithms that can compute the similarity of nodes in very large knowledge graphs, like Wikidata. To facilitate investigations and head-to-head comparisons of similarity algorithms on Wikidata, we present a user-friendly interface that allows flexible computation of similarity between Qnodes in Wikidata. At present, the similarity interface supports four algorithms, based on: graph embeddings (TransE, ComplEx), text embeddings (RoBERTa), and class-based similarity. We demonstrate the behavior of the algorithms on representative examples about semantically similar, related, and entirely unrelated entity pairs. To support anticipated applications that require efficient similarity computations, like entity linking and recommendation, we also provide a REST API that can compute most similar neighbors for any Qnode in Wikidata.

Keywords: Wikidata · Knowledge Graphs · Similarity · Embeddings

1 Introduction

Semantically *similar* entities are close in ontology space and they share essential defining properties, e.g., a car and a bus are both used for transportation and have four wheels [3]. A broader relation of semantic *relatedness* holds for entities that are dissimilar, but they are cognitively and topically close, and tend to appear in similar contexts, e.g., car and driver. Estimating similarity between two concept words have been evaluated and studied for decades [3, 10]. Much less attention has been devoted to algorithms that can compute similarity of nodes in very large knowledge graphs, like Wikidata [13]. Effective and efficient metrics of Wikidata similarity are essential for a range of downstream applications, such as entity linking [4, 5] and recommendation [6, 1].

To facilitate investigations and head-to-head comparisons of similarity algorithms on Wikidata, we present a user-friendly graphical user interface (GUI) that allows flexible computation of similarity between Qnodes in Wikidata. The similarity interface is publicly available at https://kgtk.isi.edu/similarity. At

present, the similarity GUI supports four algorithms, based on: graph embeddings(TransE [2], ComplEx [12]), text embeddings (RoBERTa [9]), and class-based similarity. Through the similarity interface, users can investigate the ability of different (families of) algorithms to capture similarity of concepts and entities in Wikidata. To support applications that require efficient similarity computations, like entity linking and recommendation, we also provide a REST API that can compute the most similar neighbors for any Qnode in Wikidata. The endpoint of this API is https://kgtk.isi.edu/nearest_neighbors.

We demonstrate the behavior of the algorithms on representative examples about semantically similar, related, or entirely unrelated entity pairs. We show that the class-based metric consistently captures semantic similarity, and assigns lower scores to terms that are merely related or unrelated. RoBERTa-based similarity behaves differently, providing high scores to both semantically similar and related pairs. The graph embedding-based metrics are somewhere in between class-based similarity and RoBERTa.

The code for the similarity GUI and our similarity API is freely available on GitHub: https://github.com/usc-isi-i2/kgtk-similarity.

2 Similarity interfaces

In this section, we describe the similarity interfaces that we have developed, together with their currently supported algorithms.

GUI Our GUI allows users to search for a primary Qnode based on its labels or aliases. The user could then add any number of secondary Qnodes in the same way, based on free text search against the node labels and aliases. We use ElasticSearch to build a text index and enable this search. The interface then displays the similarity between the primary node and each secondary Qnode, according to each of the supported algorithms.

Currently, we support four algorithms:

1. **Class similarity** computes the set of common *is-a* parents for two nodes. Here, the is-a relations are computed as a transitive closure over both the subclass-of (P279) and the instance-of (P31) relations. Each shared parents is weighted by its inverse document frequency (IDF), computed based on the number of instances that transitively belong to that parent class.
2. **TransE similarity** computes the cosine similarity between the pre-computed TransE embeddings of two Wikidata nodes.
3. **ComplEx similarity** computes the cosine similarity between the pre-computed ComplEx embeddings of two Wikidata nodes.
4. **Text similarity** computes the cosine similarity between the pre-computed RoBERTa-large embeddings of two Wikidata nodes. We pre-compute these RoBERTa embeddings over a lexicalized version of each Wikidata Qnode, based on its outgoing edges in the graph. We use outgoing edges that belong to the following seven relation types: P31 (instance of), P279 (subclass of), P106 (occupation), P39 (position held), P1382 (partially coincident with), P373 (Commons Category), and P452 (industry).

The similarity score between two nodes is computed on the fly, which allows us to facilitate estimation for any pair of Wikidata nodes. We use the operation `graph-embeddings` of the Knowledge Graph ToolKit (KGTK) [7] to compute TransE and ComplEx embeddings. We use the KGTK `text-embeddings` command to compute the text (RoBERTa-based) embeddings. A snapshot of the similarity interface is shown in Figure 1.

Nearest Neighbors API Our REST API returns K nearest neighbors for a Qnode based on the ComplEx algorithm. We index the ComplEx embeddings in a FAISS [8] index, which facilitates efficient retrieval.

3 Analysis

GUI examples In this section, we show the similarity scores provided by the supported algorithms between the Wikidata Qnode for motorcycle (Q34493) and ten other Qnodes. Specifically, we include three semantically similar nodes: bus (Q5638), Dirt Bike (Q3050907), and yacht (Q170173); four related, but dissimilar nodes: engine (Q44167), helmet (Q173603), road (Q34442), and cyclist (Q2125610); and three unrelated nodes: cheese (Q10943), Norway (Q20), and shelf (Q2637814). Following our terminology introduced in the previous section, motorcycle is the primary Qnode, and the ten additional Qnodes are secondary.

Fig. 1. Similarity between motorcycle (Q34493) and ten other terms, i.e., three semantically similar nodes: bus (Q5638), Dirt Bike (Q3050907), and yacht (Q170173); four related, but dissimilar nodes: engine (Q44167), helmet (Q173603), road (Q34442), and cyclist (Q2125610); and three unrelated nodes: cheese (Q10943), Norway (Q20), and shelf (Q2637814). The results are ordered based on their *class similarity score*.

Figure 1 shows the obtained similarity scores, in a descending order according to their class-based scores. We observe that the class-based metric consistently prioritizes semantically similar nodes over the others, as its three top-scored nodes are semantically similar to motorcycle. The remaining nodes receive notably lower scores, with the exception of the motorcycle-engine pair, whose similarity is fairly high (0.42). We observe that the class metric makes little distinction between nodes that are related and nodes that are unrelated to motorcycle. These findings show that the class metric mostly captures semantic similarity, and it does not capture semantic relatedness. This is intuitive, given that it is purely based on the Wikidata *taxonomy*, and naturally favors semantically similar terms.

Next, we order the same set of results based on their text-based score. The result is shown in Figure 2. Here, we observe that the terms that are unrelated to motorcycle (shelf, cheese, and Norway) are consistently assigned low scores. At the same time, we observe that the terms that are semantically similar (e.g., dirt bike) and merely related (e.g., cyclist) receive comparable scores. We conclude that the RoBERTa-based text similarity metric is able to discern related from unrelated nodes, but it is unable to distinguish between similar and related terms. This can be expected, considering that the RoBERTa model is trained to capture *natural language co-occurrence*, thus favoring both semantic and related terms over unrelated ones.

Fig. 2. Similarity between motorcycle (Q34493) and ten other terms, i.e., three semantically similar nodes: bus (Q5638), Dirt Bike (Q3050907), and yacht (Q170173); four related, but dissimilar nodes: engine (Q44167), helmet (Q173603), road (Q34442), and cyclist (Q2125610); and three unrelated nodes: cheese (Q10943), Norway (Q20), and shelf (Q2637814). The results are ordered based on their *text similarity score*.

Figure 3 provides a third ordering of the results, based on their TransE score. The scoring in this case correlates to a lesser extent with our a priori three-way categorization of the Qnodes, though on average semantic similarity is favored over relatedness, which is on average favored over unrelatedness. This could be explained with the property of the graph embeddings to capture *structural* similarity of nodes, i.e., to assign higher similarity between nodes that connect to similar other nodes (e.g., both engine and bus relate to car). For this reason, engine, bus, and helmet are assigned higher similarity than terms such as Norway and road.

Fig. 3. Similarity between motorcycle (Q34493) and ten other terms, i.e., three semantically similar nodes: bus (Q5638), Dirt Bike (Q3050907), and yacht (Q170173); four related, but dissimilar nodes: engine (Q44167), helmet (Q173603), road (Q34442), and cyclist (Q2125610); and three unrelated nodes: cheese (Q10943), Norway (Q20), and shelf (Q2637814). The results are ordered based on their *TransE score*.

Nearest neighbors API examples The nearest neighbors API can be leveraged to obtain the top-K most similar Wikidata Qnodes for a given Qnode. For instance, in order to obtain the top-5 most similar nodes to motorcycle (Q34493), we query: https://kgtk.isi.edu/nearest_neighbors?qnode=Q34493&k=5. The result is a list of the 5 most similar nodes, with their corresponding distance from the motorcycle Qnode and their human-readable label in English:

```
[
    {
        qnode: "Q13586807",
        score: 13.393990516662598,
        label: "Manet Korado"
```

```
    },
    {
        qnode: "Q376498",
        score: 13.482695579528809,
        label: "diesel motorcycle"
    },
    {
        qnode: "Q28126796",
        score: 15.886520385742188,
        label: "Harley-Davidson FLSTFB Fat Boy"
    },
    {
        qnode: "Q20076361",
        score: 16.452970504760742,
        label: "Honda SH50"
    },
    {
        qnode: "Q18695780",
        score: 16.553009033203125,
        label: "Bultaco TSS Mk2"
    }
]
```


Curiously, the list of the most similar nodes is dominated by specific motorcycle models and categories. The most similar three nodes are direct subclasses of the motorcycle class (connected by using the P279 relation). The remaining two Qnodes are specific models of motorcycles, represented as instances (P31) of the motorcycle class in Wikidata. This confirms our earlier observation: the graph embeddings like ComplEx assign a higher similarity to node pairs that connect to similar structures in the Wikidata graph.

4 Similarity in Downstream Tasks

Meaningful estimation of similarity is at the core of a long list of applications in natural language processing, information retrieval, and network analysis. Here, we discuss the role of estimating similarity for two prominent applications: 1) entity linking in tables, and 2) recommendation and deduplication. We also discuss how our interfaces could support these applications.

Entity linking in tables Understanding the reference of entities in tables relies on two different notions of similarity. On the one hand, entities in the same column typically are of the same type, or play the same role in a given context. For example, a table with Russian politicians will include a column with politicians, and a column with their positions. Thus, understanding entities within a column relies on similarity indicators that can capture semantic similarity, such as our class-based metric. On the other hand, entities mentioned in the same row rely on metrics that capture aspects of relatedness, such as our text-based metric, which relies on linguistic similarity, or our graph embedding metrics, which capture structural similarity. Following our previous example, this would require a

metric that can assign a high score to the pairs: Vladimir Putin - Russia, and Vladimir Putin - president.

Recommendation and deduplication A special use case of Qnode recommendation is assistance of Wikidata editors. Namely, when an editor introduces a new Qnode, it is useful to have metrics which can detect very similar existing entities and ask the editor to confirm that the new entity is different from the most similar existing ones [1]. This procedure would help to avoid introducing duplicates in Wikidata, which is a key challenge today, considering that millions of redirects have been introduced in Wikidata since its inception [11]. At the same time, similarity methods could be run over the current set of entities in Wikidata to detect potentially existing duplicates, which can be validated by an editor before their merging. The class-based metric could be used to detect potential duplicates, and it could be complemented with additional metrics (e.g., text-based similarity) when the taxonomic information is not present for a node.

5 Conclusions

This demo paper presented a user-friendly interface for computation of pairwise similarity between Qnodes in Wikidata. To facilitate head-to-head comparisons of similarity, the interface rendered the scores for multiple node pairs by four different algorithms: a class-based metric, two graph embedding metrics, and a language model based (text) metric. We experimented with their scores on semantically similar, related, or entirely unrelated entity pairs, observing that the class-based metric favored semantically similar pairs, while the text-based metric favored both semantically similar and related pairs, at the expense of the unrelated ones. Graph embeddings scored pairs orthogonally to our similarity categorization, by assigning higher scores to pairs that are structurally similar in Wikidata. To support applications where similarity plays a key role, such as entity linking, recommendation, and deduplication, we also provided a public API that returns the top-K neighbors for a given Qnode.

References

1. AlGhamdi, K., Shi, M., Simperl, E.: Learning to recommend items to wikidata editors. arXiv preprint arXiv:2107.06423 (2021)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. Advances in neural information processing systems **26** (2013)
3. Budanitsky, A., Hirst, G.: Evaluating wordnet-based measures of lexical semantic relatedness. Computational linguistics **32**(1), 13–47 (2006)
4. Cetoli, A., Bragaglia, S., O'Harney, A.D., Sloan, M., Akbari, M.: A neural approach to entity linking on wikidata. In: European conference on information retrieval. pp. 78–86. Springer (2019)
5. Delpeuch, A.: Opentapioca: Lightweight entity linking for wikidata. arXiv preprint arXiv:1904.09131 (2019)

6. Gleim, L.C., Schimassek, R., Hüser, D., Peters, M., Krämer, C., Cochez, M., Decker, S.: Schematree: Maximum-likelihood property recommendation for wikidata. In: European Semantic Web Conference. pp. 179–195. Springer (2020)
7. Ilievski, F., Garijo, D., Chalupsky, H., Divvala, N.T., Yao, Y., Rogers, C., Li, R., Liu, J., Singh, A., Schwabe, D., et al.: Kgtk: a toolkit for large knowledge graph manipulation and analysis. In: International Semantic Web Conference. pp. 278–293. Springer (2020)
8. Johnson, J., Douze, M., Jégou, H.: Billion-scale similarity search with gpus. arXiv preprint arXiv:1702.08734 (2017)
9. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692 (2019)
10. Pantel, P., Philpot, A., Hovy, E.: Matching and integration across heterogeneous data sources. In: Proceedings of the 2006 international conference on Digital government research. pp. 438–439 (2006)
11. Shenoy, K., Ilievski, F., Garijo, D., Schwabe, D., Szekely, P.: A study of the quality of wikidata. arXiv preprint arXiv:2107.00156 (2021)
12. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: International conference on machine learning. pp. 2071–2080. PMLR (2016)
13. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)