Wikidata to Bootstrap an Enterprise Knowledge Graph: How to Stay on Topic?

Lucas Jarnac[1], Pierre Monnin[1]

[1]*Orange, Belfort, France*

Abstract

An Enterprise Knowledge Graph (EKG) is a major asset for companies as it supports several downstream tasks, *e.g.*, business vocabulary sharing, search, or question answering. The bootstrap of an EKG can rely on internal business data sets or open knowledge graphs. In this paper, we consider the latter approach. We propose to build the nucleus of an EKG class hierarchy by mapping internal business terms to Wikidata entities and performing an expansion along the ontology hierarchy. This nucleus thus contains additional business terms that will, in turn, support further knowledge extraction approaches from texts or tables. However, since Wikidata contains numerous classes, there is a need to limit this expansion by pruning classes unrelated to the business topics of interest. To this aim, we propose to rely on the distance between node embeddings and node degree. We show that considering the embedding of a class as the centroid of the embeddings of its instances improves pruning and that node degree is a necessary feature to consider.

Keywords

Knowledge Graph, Pruning, Node Degree, Graph Embedding, Distance

1. Introduction

Knowledge graphs (KGs) provide a structured representation of data and knowledge in which entities are represented as nodes, and relations between them are represented as edges [1, 2]. Enterprise Knowledge Graphs (EKGs) are major assets of companies since they support various downstream applications including knowledge/vocabulary sharing and reuse, data integration, information system unification, search, or question answering [2, 3, 4]. That is why, several companies such as Google, Microsoft, Amazon, Facebook, or IBM have built their own knowledge graphs [2]. Building an EKG is an iterative and continuous process that can be carried out with various approaches. For example, it is possible to feed the KG with data and knowledge extracted from relational databases [4]. Other approaches rely on automatic knowledge extraction from semi-structured or textual data, such as the AutoKnow system for the Amazon Product Graph [5]. In such a view, it is common to first build a high quality nucleus of the KG with entities and categories extracted from premium sources. This nucleus will then support automatic knowledge extraction systems applied on a wider variety of data sources [6].

Wikidata'22: Wikidata workshop at ISWC 2022

lucas.jarnac@orange.com (L. Jarnac); pierre.monnin@orange.com (P. Monnin)

https://pmonnin.github.io (P. Monnin)

0000-0002-2819-2679 (L. Jarnac); 0000-0002-2017-8426 (P. Monnin)

In this paper, we consider this first step of building a high quality nucleus of an EKG. In particular, we focus on bootstrapping its ontology hierarchy. To this aim, we assume that we have at our disposal a repository of business terms. Building the nucleus can be achieved by mapping these terms to entities of an existing KG and integrating parts of its knowledge into the EKG. Here, we propose to perform an expansion along the ontology hierarchy of the existing KG. We retrieve the direct classes of mapped entities, their super- and sub-classes, and integrate them in the EKG (see Figure 1). Thus, the EKG nucleus contains additional business terms related to the original terms. Choosing the existing KG to integrate highly depends on the target business domains. In our work, we consider Wikidata [7], a large, free, and collaborative KG of the Wikimedia Foundation. It is a general-purpose KG that can be seen as a premium source [6]. However, due to its large size (more than 90 million items), there is a need to limit the expansion to avoid integrating unrelated terms to the target business domains and the original business terms. To illustrate, from 839 original terms related to Orange business domains, it is possible to retrieve more than 2.5 million sub-classes.

In our work, we propose to carry out this limitation by pruning unrelated classes during the expansion along the Wikidata ontology hierarchy. To do so, we rely on node degree (*i.e.*, number of incoming and outgoing edges) and the distance between node embeddings which are learned by graph embedding models. The latter encode graph structures (*e.g.*, nodes, edges) into a low-dimensional vector space that preserves as much as possible the properties of the graph [8]. Graph embeddings have shown impressive performance in various tasks such as link prediction, matching, classification, or recommendation [8, 9]. Inspired by these successful approaches, we aim at investigating in this preliminary work whether distance in the embedding space can be leveraged to prune unrelated classes. Because entities and relations are represented as low dimensional vectors, an embedding-based pruning could alleviate computational complexity issues that are one disadvantage of most pruning methods, according to Faralli *et al.* [10]. Additionally, the continuous aspect of graph embeddings compared to the discrete aspect of graph structures may allow to capture approximate relatedness [11], which would, in turn, improve pruning results. Such an approximate relatedness captured in the embedding space may also alleviate issues related to misuse of P31 ("instance of") and P279 ("subclass of") properties in the ontology hierarchy of Wikidata [12, 13, 14]. We experiment our approach on building the nucleus of an Orange Knowledge Graph based on a repository of Orange business terms [15]. In particular, we test different definitions of distance in the embedding space as well as different distance and degree thresholds. We then evaluate the quality of our pruning approach by manually labeling kept and pruned classes.

This paper is organized as follows. In Section 2, we present related work about pruning in knowledge graphs. Our pruning approach is detailed in Section 3 and evaluated on our Orange use case in Section 4. We provide a discussion of our results in Section 5 as well as future research directions in Section 6.

2. Related Work

In recent years, several methods for pruning in KGs have been proposed. Faralli *et al.* distinguish two categories of pruning: soft pruning and aggressive pruning [10]. Soft pruning requires

human input of relevant taxonomic concepts whereas aggressive pruning relies on the topology of the graph. In their paper, they propose an aggressive pruning method to extract a Direct Acyclic Graph from a potentially noisy and cyclic knowledge graph, given a set of input nodes.

Pruning approaches for KGs represent major assets to reduce the computational complexity of downstream applications, *e.g.*, recommender or question-answering systems, by allowing them to focus on relevant entities. For example, Tian *et al.* [16] propose a knowledge pruning method based on a Recurrent Graph Convolutional Network to limit information augmentation in a news recommendation perspective. A perception layer is used on each entity related to an original entity to compute the relevance between them, where only entities with a relevance larger than zero are kept. Xian *et al.* [17] integrate a user-conditional action pruning in a KG-based reinforcement learning model for a recommendation system. This pruning action relies on a scoring function that keeps the promising edges and takes into account the starting user.

To speed up a link prediction task, Joshi *et al.* [18] propose to frame it as a query-answering problem and use star-shaped subgraphs that group similar entities. In particular, they learn subgraph and query embeddings, and then compute a likelihood score between them. This score determines which subgraph will be explored to predict the correct entity by pruning irrelevant subgraphs. Regarding question-answering, Lu *et al.* [19] prune irrelevant paths with both a tail-based pruning and a path-based pruning. Tail-based pruning removes paths whose tail relation is not the relation that leads to the answer. However, this pruning is not adapted to questions involving new relations. That is why, the authors propose a path-based pruning that consists in removing paths that do not involve the domain types of all the relations in the correct path.

Alternatively, distance in the embedding space has already been leveraged in several ways. In link prediction approaches, distance can be used as a mean to measure prediction error. For example, in TransE [20], the distance between $\vec{s} + \vec{p}$ and $\vec{o}$ is used to evaluate the plausibility of a triple $\langle s; p; o \rangle$ to be true, where $\vec{s}$, $\vec{p}$, and $\vec{o}$ are the embeddings of s, p and o. Distance can also represent the semantic relatedness between entities to match [21]. Inspired by these approaches, we study in this paper whether distance in the embedding space can represent topic relatedness between classes, and thus can be used for pruning unrelated classes.

3. Expanding and Pruning from Original Business Terms

As aforementioned, we consider that we have at our disposal a set of original business terms mapped to entities of Wikidata. We then perform an expansion along the ontology hierarchy of Wikidata (Section 3.1). To keep retrieved classes related to target business domains and original business terms, this expansion is controlled with node degree and distance in the embedding space (Section 3.2).

3.1. Expansion Along the Ontology Hierarchy

Figure 1 illustrates the expansion along the hierarchy of Wikidata classes. Each original business term is mapped to a Wikidata entity represented by the starting QID. We first retrieve its direct classes, following P31 ("instance of") edges. For each of these direct classes, we then retrieve all

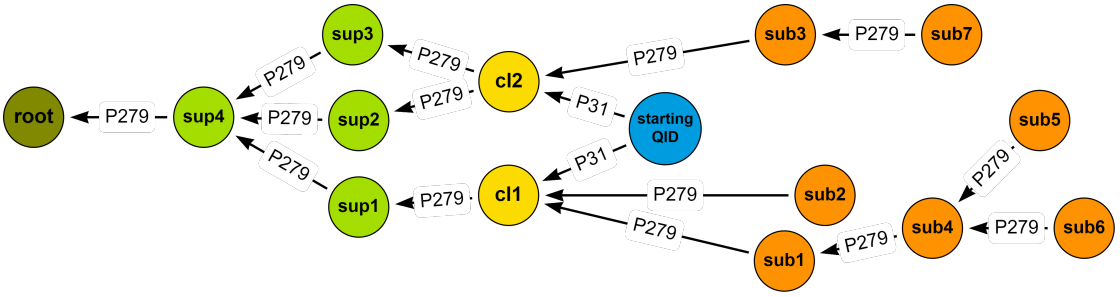


Figure 1: Illustration of the expansion along the ontology hierarchy starting from an original business term mapped to a Wikidata entity (starting QID). We first retrieve its direct classes following P31 edges (cl_i), and then their super-classes (sup_j) and sub-classes (sub_k) following P279 edges.

their super-classes by following P279 ("subclass of") edges up to the root. We also retrieve their sub-classes by following reversed P279 edges up to the deepest reachable Wikidata classes. To avoid a high number of SPARQL queries, we use a local hashmap that is built from the Wikidata dump and contains the adjacency lists of all entities in Wikidata. This allows to significantly reduce the execution time of the expansion.

3.2. Pruning Unrelated Classes

To prune unrelated classes when traversing the hierarchy of Wikidata classes, we propose to rely on node degree and distance in the embedding space. A class c that does not respect the thresholds described below is not traversed and not integrated in the EKG nucleus. This applies to direct classes, super- and sub-classes.

3.2.1. Node Degree

The degree of a node is defined as the sum of its incoming and outgoing edges. In our case, we compute the degree of Wikidata classes by only considering incoming and outgoing P31 and P279 edges since they are the only edges traversed in our expansion. For example, the degree of cl_1 in Figure 1 is equal to 4. We assume that classes with a high degree will cause a deviation from the original business domains and terms. Indeed, classes with a high degree may be too general and will lead to adding numerous super- or sub-classes to the EKG nucleus. We thus view the degree of a class as indicative of its concreteness and specificity.

To prune such classes, we use the two following thresholds. First, an absolute degree threshold $_{\text{degree-abs}}$ configured as an input parameter allows to prune classes c such that degree(c) > $_{\text{degree-abs}}$. However, this threshold may not always be applicable. Consider classes reached at a specific expansion level. It is possible for some of them to have much higher degrees than the other classes of the same level without these degrees being higher than $_{\text{degree-abs}}$. In this case, we consider these classes with relative higher degrees as anomalies. We prune them with an approach commonly used in anomaly detection. We use a relative degree threshold $_{\text{degree-rel}}$ computed at each expansion level and defined as follows:

$$_{\text{degree-rel}} = Q_3 + \quad \times (Q_3 - Q_1) \tag{1}$$

We compute the first (Q_1) and third (Q_3) quartile of the degree of classes reached at a given expansion level. The coefficient is an input parameter that allows to control how much class degree is allowed to deviate from the third quartile in terms of number of interquartile range. We then prune classes c such that degree(c) > $_{\text{degree-rel}}$. We compute this relative threshold at each expansion level since we assume degree may vary depending on the level but should be consistent at a given level. It should be noted that we apply this threshold if and only if the maximum degree at an expansion level exceeds a parameter . This parameter allows to retrieve all classes reached at an expansion level if they all have a low degree, regardless of discrepancies in their degrees.

3.2.2. Distances in the Embedding Space

We also assume that distance in the embedding space represents topic relatedness between classes. Hence, classes with high distance between their embeddings and the embedding of the starting QID may deviate from the original business domains and terms. To prune them, for each starting QID, we define a relative distance threshold $_{\text{dist-rel}}$ as follows:

$$_{\text{dist-rel}} = \quad \times \; _{\text{dist-cl}} \tag{2}$$

To compute this threshold, we first retrieve all the direct classes cl of the starting QID. We assume that these direct classes are closely related to the starting QID. Hence, they can serve as a basis to measure the remoteness of other classes in the embedding space. We compute the distances between their embeddings and the embedding of the starting QID. Direct classes whose distance with the starting QID is greater than the third quartile of distances plus the interquartile range are removed. These classes are abnormally far from the starting QID, and thus may not constitute a correct basis for remoteness. Then, we compute the mean $_{\text{dist-cl}}$ of the distances between the embeddings of the remaining direct classes and the embedding of the starting QID. In the expansion, we prune all classes c whose distance between their embeddings and the embedding of the starting QID is greater than $_{\text{dist-rel}}$. is an input coefficient that controls the allowed range of distances.

Since classes can have instances, we propose the two following definitions for the distance between a class and the starting QID:

Definition 1 (Distance D_1)**.** *The distance between a class and the starting QID is the Euclidean distance between their embeddings.*

Definition 2 (Distance D_2)**.** *The distance between a class and the starting QID is the Euclidean distance between the centroids of the embeddings of their respective instances. In case the class or the starting QID has no instance, its embedding is used instead.*

4. Experiments

We experimented our approach with the pre-trained embeddings of Wikidata available in PyTorch-BigGraph [22][1]. These embeddings were learned for more than 78,000,000 entities of the 2019-03-06 version of Wikidata.

[1] https://torchbiggraph.readthedocs.io/en/latest/pretrained_embeddings.html

4.1. Validating Distance as an Indicator of Topic Relatedness

We wanted to validate our hypothesis that distance in the embedding space can be used as an indicator of topic relatedness between classes. To this aim, we checked that the distance between embeddings of classes increases with the number of P279 edges between them. That is to say, distant classes in the ontology hierarchy should also be distant in the embedding space. The Wikidata hierarchy is known to present some issues related to misuse of P31 and P279 properties [12, 13, 14]. That is why, we expected a large range of distances between embeddings of classes separated by the same number of P279 edges. This is also the reason motivating not to rely on the number of edges for pruning.

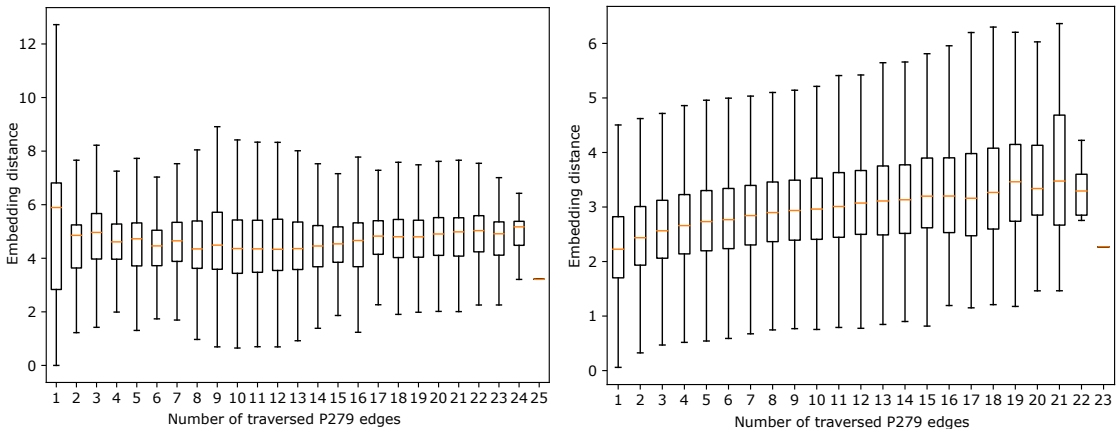
We retrieved all direct and indirect sub-classes of the root class of Wikidata (Q35120[2]). Then, for each class c, similarly to our expansion (Section 3.1), we retrieved all its super-classes and sub-classes, the number of traversed P279 edges to reach each of them, and computed the distance between their embeddings and the embedding of c. This allowed us to compute the distributions of distances between classes in the embedding space w.r.t. the number of P279 edges between them. It is noteworthy that we performed a breadth-first search to retrieve all super- and sub-classes of a class c. Consequently, only the lowest number of P279 edges between two classes was taken into account.

We conducted this experiment with the two distances presented in Section 3.2. We retrieved 2,615,380 sub-classes of the root class. Figure 2 depicts the distributions of distances obtained with 54% of these sub-classes, which accounts for around 55.8 million pairwise distances[3]. We can see that distance D_2 seems to better capture the distance in the ontology hierarchy than distance D_1. With distance D_1, the median distance oscillates whereas, with distance D_2, the median distance consistently increases with the number of P279 edges between classes. Hence, our hypothesis may only be valid with D_2. As expected, large ranges of distances exist between classes separated by the same number of P279 edges. This can be caused by misuse of the P279 property and different levels of representation granularity in different parts of the ontology hierarchy. Such a result confirms our choice not to rely on number of P279 edges between classes for pruning.

Let us illustrate the difference between the two distances outlined in Figure 2 with an example. Consider the original business term "Microsoft SharePoint" as the starting QID. We performed the expansion as described in Section 3.1 and we computed the distances between "Microsoft SharePoint" and its direct classes. Figure 3 displays these distances. We notice that, with D_1, distances in the embedding space are scattered, contrary to D_2. This leads to important discrepancies. For example, the "Content Management System" class (green circle) is a direct class and directly characterizes "Microsoft SharePoint". However, with D_1, they are far from each other in the embedding space, which could potentially lead to incorrectly prune "Content Management System". On the contrary, D_2 correctly leads to a close proximity between this class and the starting QID.

[2] https://www.wikidata.org/wiki/Q35120

[3] Our computations are limited due to time constraints.



(a) With D_1 (b) With D_2

Figure 2: Distributions of distances between embeddings of classes w.r.t. the number of P279 edges between them.

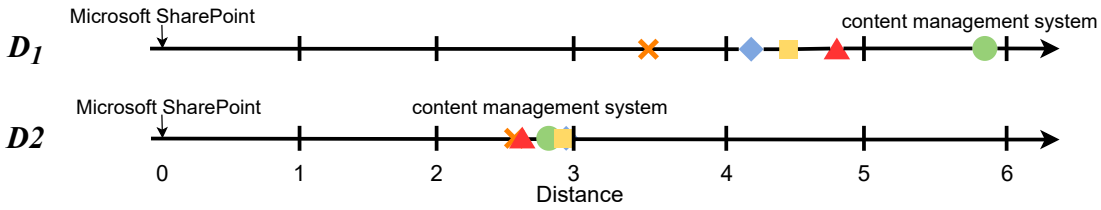
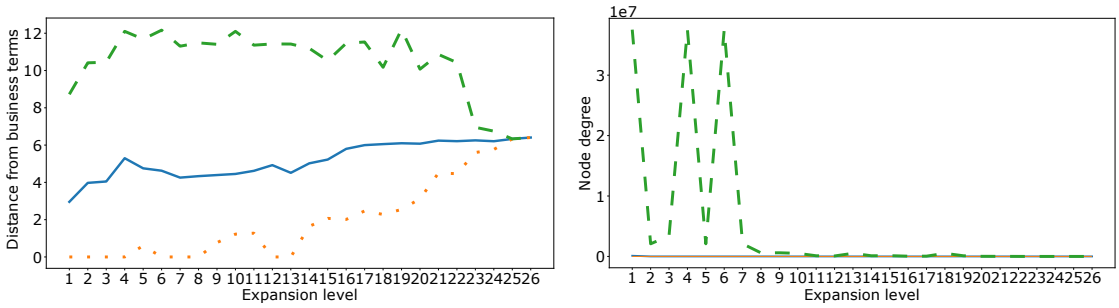


Figure 3: Distances between the original business term "Microso□ SharePoint" and its direct classes (expansion level 1): social so□ware (✕), document management system (◆), Enterprise Content Management system (■), server so□ware (▲), and content management system (●). With D_1, direct classes are dispersed. For example, "content management system" is far from the starting QID. On the contrary, D_2 leads to a better organization of distances.

4.2. Building the Nucleus of the Orange Knowledge Graph

We applied our approach to build the nucleus of the Orange Knowledge Graph. Orange is a multi-service operator (*e.g.*, telecommunications, video on demand, music, banking, cybersecurity) with more than 140,000 employees and a heterogeneous client portfolio (*e.g.*, individuals, companies). To face such a diversity of activities, an internal repository contains public and internal business terms together with their textual definitions.

839 business terms were manually matched with their corresponding Wikidata entities before applying the expansion mechanism described in Section 3.1. This matching was manually performed by AI researchers from Orange who identified "same as" alignments between terms in the internal repository and Wikidata entities. To help them in their task, each term was presented with candidate entities based on string similarity between terms and labels of entities. Their expansion led to retrieve 393 distinct direct classes, 946 distinct super-classes, and 2,560,426 distinct sub-classes. From these results, we chose to focus only on pruning sub-classes since their important number may indicate unrelated classes to the original business terms.



(a) Distances (D_2 definition) (b) Node degrees

Figure 4: Maximum (- -), mean (—), and minimum (···) distances and node degrees of retrieved classes per level of expansion from 839 original business terms.

To illustrate the evolution of the two characteristics of interest of our pruning approach, we show in Figure 4a the maximum, mean, and minimum distances (with the D_2 definition) of reached classes at each level of expansion from our 839 starting QIDs. Similarly, Figure 4b depicts the maximum, mean, and minimum degrees of reached classes at each level of expansion from our 839 starting QIDs. We notice that the mean distance only increases slightly through the expansion w.r.t. its initial value at expansion level 1. Such a stability makes us think that distances of direct classes are good representatives of close distances and can effectively serve as a basis to evaluate remoteness in the embedding space. This validates our definition of $_{\text{dist-rel}}$ (Equation (2)) as dependent upon the mean distance of direct classes. Figure 4b illustrates that many generic classes are retrieved. For example, “Galaxy”, “Human”, “Taxon”, and “Star” are retrieved from “Linux”. However, sub-classes retrieved at the next expansion level from such generic classes may be unrelated to the original business terms. Interestingly, these peaks in node degrees are not associated with peaks in distances. This observation confirms the need for a degree-based pruning beside a distance-based pruning. In particular, the fixed degree threshold allows to tackle the degree peaks displayed.

We also performed six expansion and pruning experiments from the 839 original business terms with different configurations. We fixed $_{\text{degree-abs}} = 200$, $= 1.5$, and $= 20$ but tested with different values of $2\ f1.2; 1.25; 1.3g$ and the two definitions of distance D_1 and D_2. We manually labeled the pruned and kept classes to evaluate the performance of our approach. Results are presented in Table 1. Each row presents the results of one experiment in which all defined thresholds are applied. Since configurations differ between expansions, classes may be differently pruned in some expansions due to distance thresholds. This leads to different explorations of the ontology hierarchy, and thus different numbers for degree-based pruning. The left part of Table 1 presents the number of pruned classes and the resulting precision per pruning threshold as well as global results. It should be noted that some classes were pruned by exceeding both the $_{\text{degree-rel}}$ and $_{\text{dist-rel}}$ thresholds (represented in columns (4)). The right part of Table 1 presents the number of kept classes and the resulting precision. Results are discussed below.

Table 1
Number and precision of pruned and kept classes with the two distances and di□erent configurations for . (1) stands for pruning with $_{\text{degree-rel}}$; (2) stands for pruning with $_{\text{degree-abs}}$; (3) stands for pruning with $_{\text{dist-rel}}$; (4) stands for pruning with both $_{\text{degree-rel}}$ and $_{\text{dist-rel}}$; (5) stands for global pruning. Values in bold indicate the best precision.

		# Pruned classes					Precision					# Kept classes	Precision
		(1)	(2)	(3)	(4)	(5)	(1)	(2)	(3)	(4)	(5)		
D_1	1.2	11	45	1,344	182	1,582	**0.91**	**0.87**	0.69	0.78	0.71	1,135	**0.86**
	1.25	19	46	1,289	183	1,537	0.89	**0.87**	0.73	0.79	0.74	1,293	0.83
	1.3	25	46	1,224	189	1,484	0.88	**0.87**	0.75	0.79	0.76	1,484	0.77
D_2	1.2	184	60	2,108	118	2,470	0.74	0.85	0.75	0.92	0.76	1,645	0.81
	1.25	213	64	2,032	110	2,419	0.77	0.84	0.79	0.92	0.79	1,931	0.76
	1.3	250	65	1,917	100	2,332	0.79	0.83	**0.84**	**0.94**	**0.84**	2,311	0.71

5. Discussion

Regarding pruning precision, we note in Table 1 that D_2 obtains a better precision than D_1 for distance-based pruning. This result was expected given the better organization of the embedding space with D_2 than with D_1, as described in Section 4.1. It appears through our experiments that a distance based on centroids of class instances seems to better carry the relatedness between classes. D_2 may thus better cope with "instance of" and "subclass of" properties not always being used correctly in the hierarchy which impacts the quality of the Wikidata ontology [12, 13, 14]. Table 1 shows that D_1 reaches a better precision for degree-based pruning but this only applies to a reduced number of classes. Consequently, the better performance of D_1 on these thresholds has little impact on global pruning precision, whose best result is obtained with D_2.

We notice that more classes are pruned with D_2 than with D_1. Similarly, more classes are kept with D_2. This indicates that D_2 leads to a different and more extensive hierarchical exploration than D_1. This more extensive exploration is at the expanse of the precision of kept classes that is lower with D_2 in all configurations. Conversely, the more restrained exploration with D_1 comes at the expanse of the precision of pruned classes that is lower in all configurations. However, precision of kept classes with D_2 is only lower by 5 to 7 points for 510 to 827 additionally kept classes. This means that many of the additionally kept classes are correct. Hence, D_2 appears to be a better setting that offers a further enrichment of the EKG nucleus without an important precision decrease. In both cases, our pruning approach allows to reduce the number of integrated classes in the EKG nucleus from more than 2.5 million (without pruning) to around 2,000 with a good precision.

Regarding distance in the embedding space, it should be noted that requiring a user to provide a distance threshold is not a trivial task since such a distance may not be interpretable. This assessment is in line with the need for semantic embeddings to give a meaning and an interpretation to distance in the embedding space to humans [23]. In this perspective, Wikidata could serve as a benchmark. It would also be interesting to validate our hypothesis of topic-

relatedness representativity not only with super- or sub-classes as in Section 4 but with siblings that are not currently considered in our approach.

Table 1 shows that precisions of pruned and kept classes are inter-dependent. Indeed, increasing the distance coefficient improves the precision of pruned classes at the expanse of the precision of kept classes. This means that a higher distance threshold allows to remove less classes but with a higher precision. However, some of the additionally retrieved classes are unrelated to the original business terms. This shows that the decision boundary between classes to prune and to keep is not perfectly captured with our approach and the two considered features, *i.e.*, node degree and distance in the embedding space. A trade-off must thus be found to obtain good precision for both pruned and kept classes.

One can wonder about the performance of other embedding models for a pruning based on Euclidean distance since they may differently organize the embedding space. A future work on performance comparison should also consider symbolic pruning approaches and take into account scalability and execution time beside precision. Going a step further, we could envision a machine learning model that learns or adapts graph embeddings to better prune classes. Such a model could, for example, use Graph Convolutional Networks (GCN) with the Soft Nearest Neighbor Loss (SNNL) as in [21]. To reduce user input, only pruned classes could be labeled which would lead to a semi-supervised pruning task. Finally, to further expand the EKG nucleus, instances of kept classes could also be evaluated with our thresholds to be integrated.

6. Conclusion

In this paper, we aimed at building a nucleus of the class hierarchy of an Enterprise Knowledge Graph for Orange. To this aim, we mapped original business terms available in an internal repository to their corresponding entities in Wikidata. Then, we performed a hierarchical expansion along the ontology hierarchy to integrate super- and sub-classes. Due to the high number of traversed classes, we proposed to limit this expansion with a pruning approach relying on node degree and distance in the embedding space. Our experiments showed that considering the embedding of a class as the centroid of the embeddings of its instances improves distance pruning. Results also highlighted that node degree is an effective and necessary feature that cannot be substituted by distance in the embedding space. In future works, we ambition to confirm these results with different graph embedding models and investigate the learning of graph embeddings specific to the pruning task. The EKG nucleus could also be further enriched by matching Orange with similar corporations and retrieving their Wikidata statements. Such statements could also undergo a pruning process similar to the one described in this paper.

References

[1] A. Hogan, et al., Knowledge Graphs, Synthesis Lectures on Data, Semantics, and Knowledge, Morgan & Claypool Publishers, 2021.

[2] N. F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, J. Taylor, Industry-scale knowledge graphs: lessons and challenges, Commun. ACM 62 (2019) 36–43.

[3] M. Galkin, S. Auer, S. Scerri, Enterprise knowledge graphs: A backbone of linked enterprise data, in: 2016 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2016, Omaha, NE, USA, October 13-16, 2016, IEEE Computer Society, 2016, pp. 497–502.
[4] J. Sequeda, O. Lassila, Designing and Building Enterprise Knowledge Graphs, Synthesis Lectures on Data, Semantics, and Knowledge, Morgan & Claypool Publishers, 2021.
[5] X. L. Dong, et al., AutoKnow: Self-driving knowledge collection for products of thousands of types, in: KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020, ACM, 2020, pp. 2724–2734.
[6] G. Weikum, X. L. Dong, S. Razniewski, F. M. Suchanek, Machine knowledge: Creation and curation of comprehensive knowledge bases, Foundations and Trends Databases 10 (2021) 108–490.
[7] D. Vrandecic, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Communications of the ACM 57 (2014) 78–85.
[8] H. Cai, V. W. Zheng, K. C. Chang, A comprehensive survey of graph embedding: Problems, techniques, and applications, IEEE Trans. Knowl. Data Eng. 30 (2018) 1616–1637.
[9] S. Ji, S. Pan, E. Cambria, P. Marttinen, P. S. Yu, A survey on knowledge graphs: Representation, acquisition, and applications, IEEE Transactions on Neural Networks and Learning Systems 33 (2022) 494–514.
[10] S. Faralli, I. Finocchi, S. P. Ponzetto, P. Velardi, Efficient pruning of large knowledge graphs, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden, ijcai.org, 2018, pp. 4055–4063.
[11] R. V. Guha, Towards A model theory for distributed representations, in: 2015 AAAI Spring Symposia, Stanford University, Palo Alto, California, USA, March 22-25, 2015, AAAI Press, 2015.
[12] F. Brasileiro, J. P. A. Almeida, V. A. de Carvalho, G. Guizzardi, Applying a multi-level modeling theory to assess taxonomic hierarchies in Wikidata, in: Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11-15, 2016, Companion Volume, ACM, 2016, pp. 975–980.
[13] A. Piscopo, C. Phethean, E. Simperl, What Makes a Good Collaborative Knowledge Graph: Group Composition and Quality in Wikidata, in: Social Informatics - 9th International Conference, SocInfo 2017, Oxford, UK, September 13-15, 2017, Proceedings, Part I, volume 10539 of *Lecture Notes in Computer Science*, Springer, 2017, pp. 305–322.
[14] K. Shenoy, F. Ilievski, D. Garijo, D. Schwabe, P. A. Szekely, A study of the quality of Wikidata, Journal of Web Semantics 72 (2022) 100679.
[15] Y. Chabot, P. Monnin, F. Deuzé, V. Huynh, T. Labbé, J. Liu, R. Troncy, A framework for automatically interpreting tabular data at orange, in: Proceedings of the ISWC 2021 Posters, Demos and Industry Tracks: From Novel Ideas to Industrial Practice co-located with 20th International Semantic Web Conference (ISWC 2021), Virtual Conference, October 24-28, 2021, volume 2980 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2021.
[16] Y. Tian, Y. Yang, X. Ren, P. Wang, F. Wu, Q. Wang, C. Li, Joint knowledge pruning and recurrent graph convolution for news recommendation, in: SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021, ACM, 2021, pp. 51–60.

[17] Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, Y. Zhang, Reinforcement knowledge graph reasoning for explainable recommendation, in: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019, ACM, 2019, pp. 285–294.

[18] U. Joshi, J. Urbani, Searching for embeddings in a haystack: Link prediction on knowledge graphs with subgraph pruning, in: WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020, ACM / IW3C2, 2020, pp. 2817–2823.

[19] J. Lu, Z. Zhang, X. Yang, J. Feng, Efficient subgraph pruning & embedding for multi-relation QA over knowledge graph, in: International Joint Conference on Neural Networks, IJCNN 2021, Shenzhen, China, July 18-22, 2021, IEEE, 2021, pp. 1–8.

[20] A. Bordes, N. Usunier, A. García-Durán, J. Weston, O. Yakhnenko, Translating embeddings for modeling multi-relational data, in: Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States, 2013, pp. 2787–2795.

[21] P. Monnin, C. Raïssi, A. Napoli, A. Coulet, Discovering alignment relations with graph convolutional networks: A biomedical case study, Semantic Web 13 (2022) 379–398.

[22] A. Lerer, L. Wu, J. Shen, T. Lacroix, L. Wehrstedt, A. Bose, A. Peysakhovich, Pytorch-biggraph: A large scale graph embedding system, in: Proceedings of Machine Learning and Systems 2019, MLSys 2019, Stanford, CA, USA, March 31 - April 2, 2019, mlsys.org, 2019.

[23] H. Paulheim, Make embeddings semantic again!, in: Proceedings of the ISWC 2018 Posters & Demonstrations, Industry and Blue Sky Ideas Tracks co-located with 17th International Semantic Web Conference (ISWC 2018), Monterey, USA, October 8th to 12th, 2018, volume 2180 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2018.