Entity Typing based on RDF2Vec using Supervised and Unsupervised Methods

Radina Sofronova[1,2]✉, Russa Biswas[1,2]✉, Mehwish Alam[1,2], and Harald Sack[1,2]

[1] FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Germany
[2] Karlsruhe Institute of Technology, Institute AIFB, Germany
radina.sofronova@student.kit.edu,
{russa.biswas,mehwish.alam,harald.sack}@fiz-karlsruhe.de

Abstract. Knowledge Graphs have been recognized as the foundation for diverse applications in the field of data mining, information retrieval, and natural language processing. So the completeness and the correctness of the KGs are of high importance. The type information of the entities in a KG, is one of the most vital facts. However, it has been observed that type information is often noisy or incomplete. In this work, the task of fine-grained entity typing is addressed by exploiting the pre-trained RDF2Vec vectors using supervised and unsupervised approaches.

1 Introduction

Entity typing is the process of assigning a type to an entity which is a fundamental task in Knowledge Graph (KG) construction and completion. Most of these KGs are created either via automated information extraction from Wikipedia infoboxes, information accumulation provided by the users, or by using heuristics. Therefore, one primary problem is that majority of the entities have coarse-grained type information. The classes in DBpedia have a hierarchical structure, in which the classes in the same branch of hierarchy share common characteristic features. A class retains the basic characteristics of the parent as well as has some unique features of its own. Fine-grained type prediction is the task of assigning a type or class to an entity.For example, in DBpedia, the actor *dbr: Tom_Hanks* is of type *dbo: Person*, whereas the most appropriate class, would be *dbo: Actor*, which is a subclass of *dbo: Person*. Table 1 depicts a brief statistics of the missing fine-grained types in DBpedia. For instance, class *dbo: SportsTeam* has 14 subclasses in DBpedia and 352006 entities, out of which only 8.9% are assigned to its subclasses. To deal with this challenge, a few approaches have already been proposed. The first approach is based on statistical heuristics [6] and the second approach uses a supervised hierarchical classification [4, 5]. On the other hand, [2] considers the text as well as the structural information in KGs for entity typing. In [1], the authors use the abstracts from the Wikipedia pages to predict the Wikipedia infobox types using word embeddings. Since DBpedia type information is extracted from the Wikipedia infobox types, these

Table 1: Distribution of entities in some of the subclasses in DBpedia.

Class	#Entities	#Coarse-Grained Typed entities	%Fine-Grained Typed Entities
SportsTeam	352006	320835	8.9%
Company	70208	55524	20.9%
Settlement	478906	246163	48.6%
Activity	19464	8824	54.7%
Event	76029	19418	74.5%

models can be used for entity typing. However, none of these approaches have exploited KG embeddings to solve the problem of type prediction. This work focuses on evaluating the role of KG embeddings only in assigning fine-grained type information to entities in DBpedia that already have a coarse-grained type.

2 Supervised and Unsupervised Entity Typing

RDF2Vec. RDF2Vec [7] generates latent representations of entities in a KG into a lower dimensional feature space. The embedding space comprises of entities and properties, in which the semantically similar entities are closely spaced. In this work, the pre-trained DBpedia embeddings generated using RDF2Vec have been used. The original study uses graph walks and word2vec for generating the embeddings of DBpedia.

Unsupervised Approach - Vector Similarity. In order to assign fine-grained type to an entity with an already assigned coarse-grained type, class hierarchy in DBpedia has been exploited. Cosine similarity between the entity vector and all the class vectors of the classes in that branch of the class hierarchy in DBpedia were calculated. For example, in DBpedia, for the entity *dbr:Baker&McKenzie*, the `rdf:type` class is *dbo:LawFirm*. Next, class hierarchy of *dbo:LawFirm* is traversed to find the highest level parent class dbo:Organisation after *dbo:Agent*. Now, all the subclasses of dbo:Organisation in the hierarchy are extracted and the cosine similarity between all the subclasses and the entity *dbr:Baker&McKenzie* has been calculated.

The approach has been explored with two alternatives for the class vectors. The first uses the pre-trained RDF2Vec vectors of the classes for the cosine similarity. Since, a set is represented by its members, which exhibit the same properties. Similarly, the entities of the classes in DBpedia have the same or similar properties. Therefore, following this concept, the average of the entity vectors of a class is a representation of the class in DBpedia and is given by,

$$class\_vector = \frac{1}{n}(\mathbf{v_1} + \mathbf{v_2} + ... + \mathbf{v_n}), \tag{1}$$

where n is the number of entities in a class and the $\mathbf{v_i}, i \in [1, n]$ are the RDF2Vec vectors of all entities of the class. This class vector is used for the vector similarity in the second approach.

Table 2: Accuracy of Vector Similarity and CNN for different datasets

Datasets	RDF2Vec				
	Vector Similarity				CNN
	Original Class vector		Class Vector as avg. of entity vectors		
	Hits@3	Hits@1	Hits@3	Hits@1	
59 classes, 500 entities/class	68.91%	21.63%	**95.63%**	86.51%	81.78%
86 classes, 2k entities/class	34.74%	12.26%	**84.78%**	74.9%	53.67%
81 classes, 4k entities/class	32.51%	11.01%	**84.61%**	74.33%	53.49%

Supervised Approach - 1D CNN. The entity typing problem is converted to a classification problem with the `rdf:type` as classes in which, a 1D CNN model [3] is built on top of RDF2Vec. The model consists of a convolutional layer which involves a feature detector followed by a global max pool layer. For regularization, a dropout on the output of the pooling layer is used which is then passed through a fully connected final layer.

3 Experiments and Results

Dataset. In order to have fine-grained type prediction of the entities which are already coarse-grained typed in DBpedia 2016-10[3], a dataset with 59 classes considering the type hierarchy was generated. Moreover, the selected classes are less popular i.e., 15 classes have less than 500 entities per class, 20 classes have entities between 500 and 1000, and the remaining 24 classes have more than 1000 entities. From each of the chosen classes, 500 entities were extracted. The other datasets contain 86 classes with 2000 entities per class and 81 classes with 4000 entities per class respectively. Therefore, this study provides a basic analysis of the quality of vectors for lesser known classes and entities and their performance in entity typing task. The uniform vectors in RDF2Vec model[4] with a dimension of 200, have been used.

Experiments.
Unsupervised Approach. Hits@1 and Hits@3 have been computed on the values of vector similarity compared against the current entity types in DBpedia. The vector similarity values for a certain entity are ranked to determine if the correct entity type is present in top 1 and 3 in the list of types. Table 2 shows that the experiments depict that RDF2Vec pre-trained class vectors do not reflect the characteristics of the entities of the class. This is due to the fact that RDF2Vec is path dependent and considers only the outgoing edges in the

[3] https://wiki.dbpedia.org/downloads-2016-10
[4] https://zenodo.org/record/1320211#.Xbnwf25FydI

Table 3: Comparison with SDType model

Datasets	#Common Entities	Accuracy		
		SDType	RDF2Vec Vector Similarity	
			Hits@1	Hits@3
59 classes, 500 entities/class	7425	**82.35%**	73.24%	79.5%
86 classes, 2k entities/class	57467	80.43%	74.44%	**81.2%**
81 classes, 4k entities/class	109948	81.22%	75.12%	**82%**

Table 4: Accuracy of 1D CNN models.

CNN models	20 classes, 150 entities/class	59 classes, 500 entities/class
1 1D layer	59.83%	42.61%
2 1D layers	93.5%	81.78%

RDF graph. In contrast, the class vectors generated by the average of the entity vectors are able to reflect the characteristics of a class. It is observed that with this approach, the best results of entity typing is achieved with this method in Hits@3. The detailed results and plots are available online[5]. Moreover, the unsupervised model is compared with the statistical heuristic based entity typing approach [6] in Table 3. For this, the publicly available results of SDType method[6] have been used. However, only a small fraction of the entities are common between the available results and our datasets as depicted in the second column of Table 3. Therefore, a comparison with the whole dataset is not possible. The accuracy provided in this table is calculated based on the number of common entities. It is to be noted that KG embedding based vector similarity method works better than SDType for two of the datasets.

Supervised Approach. Two approaches have been followed here. The first model consists of two 1D CNN layers and was trained with 150 entities for 20 classes each with a 80-20 split of training and test set. Since the number of entities per class is small, the model is overfitting with two convolutional layers yielding 93.5% accuracy as shown in Table 4. The two basic ways of reducing overfitting in neural network models are training the network on more data or by changing the complexity of the network, have been examined. In the second experiment, 59 classes with 500 entities were taken which results in 81.78% accuracy. Also, the complexity of the model is reduced to only one convolutional layer which results in 42.61% accuracy for 59 classes. It can be concluded from the results that a CNN model on the top of the RDF2Vec works better for entity

[5] https://github.com/ISE-FIZKarlsruhe/DBpedia-Entity-Typing-with-RDF2Vec
[6] http://downloads.dbpedia.org/2016-10/core-i18n/en/instance_types_sdtyped_dbo_en.ttl.bz2

typing. Moreover, reducing the complexity of model works better for the smaller datasets. For the other two datasets, it has been observed that the accuracy does not affect much with the increase in the dataset size and reducing the number of classes. Overall, it is to be noted that unsupervised vector similarity approach with the averaged class vector works best for all the datasets.

4 Conclusion and Future Work

In this paper, different approaches for entity typing in a KG have been analyzed. The achieved results using the set theory concept when applied to generate the class vectors from the pre-trained entity vectors proved to be the best approach for the task with this embedding model. On the other hand, the pre-trained RDF2Vec vectors coupled with CNN work second best. The method can be adapted to any other KG with any pre-trained embedding model. In future work, the vectors from the graph kernel method of RDF2Vec for entity typing will be explored as well as more information from the KG. Moreover, contextual word embeddings will also be exploited on the abstract of the entities coupled with KG embeddings to improve the entity type prediction.

References

1. Biswas, R., Türker, R., Moghaddam, F.B., Koutraki, M., Sack, H.: Wikipedia infobox type prediction using embeddings. In: Proceedings of the 1st Workshop on Deep Learning for Knowledge Graphs and Semantic Technologies DL4KGS 2018. CEUR Workshop Proceedings (2018)
2. Jin, H., Hou, L., Li, J., Dong, T.: Attributed and predictive entity embedding for fine-grained entity typing in knowledge bases. In: Proceedings of the 27th International Conference on Computational Linguistics. Association for Computational Linguistics (2018)
3. Kim, Y.: Convolutional neural networks for sentence classification. In: Proceedings of the 2014 Conference on EMNLP. Association for Computational Linguistics (2014)
4. Kliegr, T., Zamazal, O.: Lhd 2.0: A text mining approach to typing entities in knowledge graphs. Journal of Web Semantics (2016)
5. Melo, A., Paulheim, H., Völker, J.: Type prediction in rdf knowledge bases using hierarchical multilabel classification. In: Proceedings of the 6th International Conference on WIMS 2016. Association for Computing Machinery (2016)
6. Paulheim, H., Bizer, C.: Type inference on noisy rdf data. In: The Semantic Web – ISWC 2013. Springer Berlin Heidelberg (2013)
7. Ristoski, P., Paulheim, H.: Rdf2vec: Rdf graph embeddings for data mining. In: The Semantic Web - ISWC 2016 : 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I. Springer International Publishing (2016)