Matching Words and Knowledge Graph Entities with Meta-Embeddings

Damien Sileo[1], Camille Pradel[1], Guillermo Eschegoye[2], Anselmo Peñs[2], Arantxa Otgi[4], Jan Milan Driu[3], Mark Cielebak[3], Ander Barena[4], et Eneko Agirre[4]

[1]Synapse Développement,Toulouse
[2]Universidad Nacional de Educación a Distanca, Madrid
[3]Zurich University of Applied Scienes, Zurich
[4]IXA NLP Group, University of the Basque Country, Donostia

31 mai 2019

Résumé

Word vectors are a key component for matching distinct textual units semantically. However, they are not directly applicable for matching text with structured data, for which graph embeddings exist. In this work, we propose a flexible method in order to map the representation of graph embeddings to word embeddings representation. Thus, we can improve word embeddings with a weighted average with mapped graph embeddings. We evaluate our models on the task of matching natural language questions and SPARQL queries, and significantly improve queries matching accuracy. We also evaluate word meta-embeddings intrinsically and show improvements over previous models.

Mots-clef : Question answering, Knowledge Graphs, Word Embeddings, Graph Embeddings, Meta-Embedding

1 Introduction

Structured data has become ubiquitous, abundant and involved in numerous applications. Knowledge bases like DBpedia, Wikidata, OpenCyc [FEMR15] provide large and growing structured resources. They contain millions of facts represented as triplets such as (Paris, LOCATED_IN, France). Formal languages such as SPARQL and scalable endpoint architectures allow efficient queries. However, natural language is more convenient for most users. Translating natural language queries into formal language queries (e.g. SPARQL) has been a long standing artificial intelligence task. Table 1 shows an example of a natural language question with associated SPARQL query. But current systems are only successful on restricted versions of this task, e.g. using specific patterns [PHH13, TMDL17].

Since full translation-based systems are not reliable, a useful task would be the matching of related SPARQL requests (either from historical data or from the output of a translation-based system) according to their similarity to a natural language question. In this paper, we tackle the prediction of similarity between natural language questions and SPARQL requests. Word embeddings are a key component in many textual similarity systems and have been used to represent natural language questions. However, the components of SPARQL queries are either SPARQL keywords (e.g. SELECT) or Uniform Ressource Unifiers (URI) (e.g. http ://dbpedia.org/resource/Stanley_Kubrick).

There exists pre-computed URI embeddings, but learning an alignment of the embeddings latent space is needed for similarity computations, and relying on task specific manually annotated data is costly. Meta-embeddings could be used in order to solve this problem. A meta-embedding is a representation derived from a set of distinct embeddings (e.g. Word2Vec and GloVe). Yet, there exists no meta-embedding method leveraging pretrained knowledge graph embeddings and word embeddings. Such meta embedding could also allow integration of symbolic external knowledge for common sense reasoning or information retrieval.

Our contributions are as follows :

— A meta-embedding method to align word embeddings with graph entities embeddings

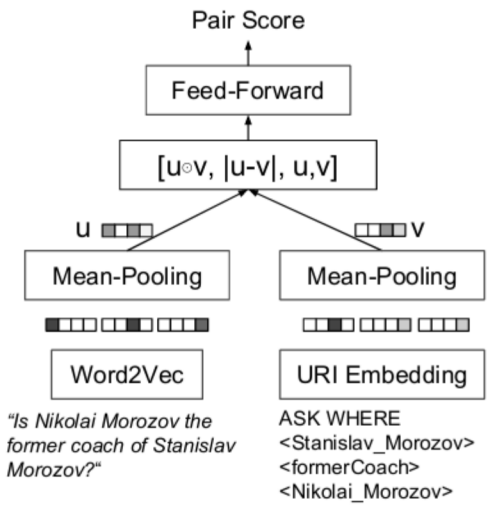

FIGURE 1 – Model architecture for similarity estimation

— Experiments on Natural Language/SPARQL queries similarity prediction
— Intrinsic evaluation of our meta-embeddings

2 Similarity estimation models

A possible use-case of our method is improvement on similarity prediction between a natural language question and a SPARQL query. In this work, we use a siamese neural network whose architecture is depicted in the figure 1 for such similarity estimation in a supervised setup. We experimented using state of the art [BSA18] models but they did not perform well, probably due to the short contexts of queries, as opposed to the evaluation datasets used in entity linking literature.

We represent the questions/queries with the average of its symbols (words/URI) embeddings, composed with a matching function (with a concatenation of hadamard product, absolute difference of input vectors) followed by a feed-forward neural network. Average-pooling is a simplistic sequence encoding method but it was shown to be competitive with more complicated architectures [SWW+18].

Representing words from natural language questions is straightforward using word embeddings. By contrast, there are several ways to represent DBPedia URI (e.g. http ://dbpedia.org/resource/Stanley_Kubrick). For instance, text can be derived from the label for the URI (e.g *Stanley Kubrick*) allowing the use of word embeddings but disregarding the knowledge from the DBPedia graph.

Pre-computed DBPedia URI embeddings [RP16] can also be used. They are embeddings computed with the SkipGram algorithm (used in Word2Vec and Node2Vec []) with DBPedia [1] graph walks instead of sentences. Such graph walks encode knowledge about entities. For example,

(Stanley_Kubrick,
writer,
A_Clockwork_Orange)

is a possible sub-path containing some useful information about Stanley Kubrick.

RDF2Vec inherits many properties from Word2Vec vectors (e.g. a cosine similarity that reflect relatedness).

In this work, we will compare the use of RDF2Vec and Word2Vec for URI representation, and propose a meta-embedding method to combine them.

3 Proposed Meta-Embedding

Word and graph embeddings encode complementary knowledge, but their latent space need to be aligned in order to perform similarity computations. Here, we propose to learn to map the latent space of RDF2Vec to the space of word embeddings.

To do so, we train a feed-forward neural network f_θ in order to predict the word embedding Word2Vec(u) representation of a given URI u from its URI embedding RDF2Vec(u). More specifically, we optimize θ in the following loss function :

$$\mathcal{L} = \sum_{u \in \mathcal{V}} \text{MSE}(\text{Word2Vec}(u), f_\theta(\text{RDF2Vec}(u))) \quad (1)$$

$\mathcal{V}$ is the set of training examples, i.e. the set of URIs where a word in a label matches a word vector. When several words are found, we use the average of their embeddings. Figure 2 illustrates this approach.

As $f_\theta(\text{RDF2Vec}(u))$ is trained to lie in the same space as Word2Vec(u), a weighted average of these representations can be also used :

$$\text{Weighted}_\alpha(u) = (1-\alpha)\text{Word2Vec}(u) + \alpha f_\theta(\text{RDF2Vec}(u)) \quad (2)$$

1. (2016-04 version)

$\text{question}_{\text{NL}}$	How many movies did Stanley Kubrick direct ?
$\text{query}_{\text{SPARQL}}$	SELECT DISTINCT COUNT(?uri) WHERE ?uri <http ://dbpedia.org/ontology/director> <http ://dbpedia.org/resource/Stanley_Kubrick>

TABLE 1 – Sample from *LC-Quad* dataset

URI Representation	Cross-Entropy	Accuracy (%)
None (Majority Class Prediction)	0.6931	90.00
Word2Vec	0.1262	97.81
g_W(RDF2vec)	0.2595	95.06
f_θ(R2Vec)	0.2610	90.57
Weighted ($\alpha = 0.075$)	**0.1189**	**97.94**

TABLE 2 – Test results of different URI embeddings models ; bold value denotes the best results *Weighted* is defined in equation 2

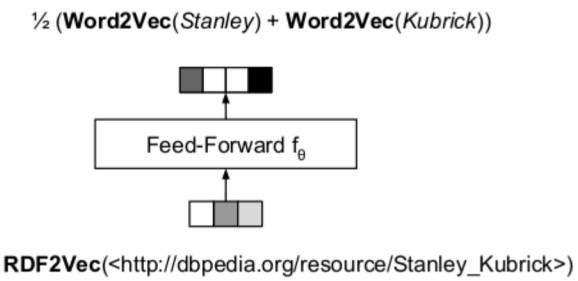

FIGURE 2 – Model architecture for embedding alignment

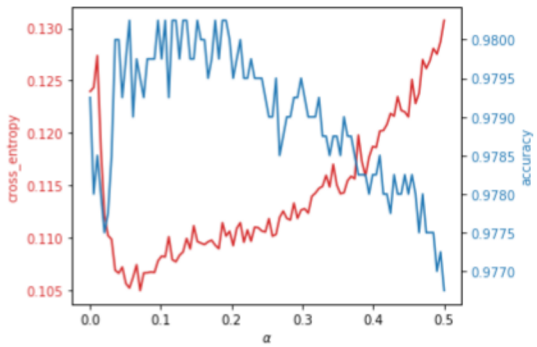

FIGURE 3 – Influence of α on matching prediction validation results

4 Experiments

4.1 Query matching evaluation

We evaluate our models on the LC-QuAD [TMDL17] dataset which is a collection of 5000 natural language questions with associated SPARQL queries. 4000 pairs are used for training and the remaining is used for evaluation. For each example, we generate 9 examples of dissimilar NL/SPARQL queries using random associations of different queries. This process is done on train data and test data separately.

To represent natural language questions, we always use word embeddings from [MGB+18] trained on CommonCrawl.

Regarding URI representations, we evaluate several embeddings :

g_W(RDF2Vec) is a linear projection of RDF2Vec embedding. The projection W is initialized randomly and learnt during the matching prediction training.

f_θ is instanciated with a two hidden layer MLP (hidden layer sizes are 200,200) with batch-normalization and ReLu activation. θ is trained on $6.0M$ URIS, using 1 epoch and using Adam optimizer [KB15] with default parameters, using the loss from equation 1. The pararameters are kept fixed in the matching prediction training.

For the matching detection training, 10% of training data is kept aside as validation set in order to determine the best number of epochs (found to be 8, also using Adam optimizer).

We performed cross validation on the parameter α. Figure 3 shows the influence on α on evaluation metrics. $\alpha = 0$ is the same as only using Word2Vec, and $\alpha = 1$ is equivalent to only using f_θ(RDF2Vec).

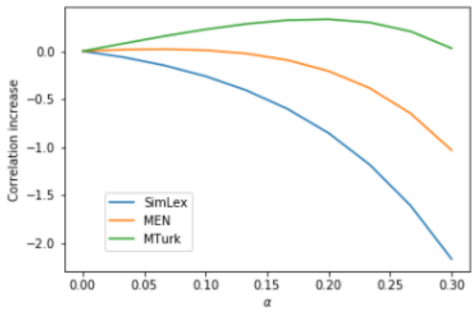

FIGURE 4 – Influence of α on word similarity prediction evaluation using the weighted combination of Word2Vec and f_θ(RDF2Vec). y axis is the pearson correlation improvement over the Word2Vec baseline.

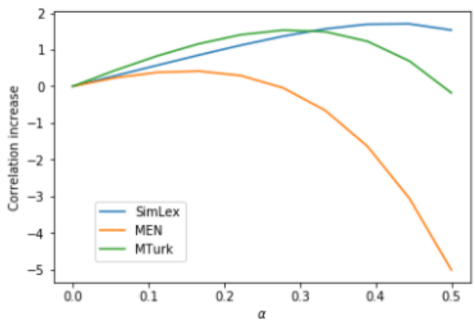

FIGURE 5 – Influence of α on word similarity prediction evaluation using the weighted combination of Word2Vec and f_θ(WordNet2Vec). y axis is the pearson correlation improvement over the Word2Vec baseline.

4.1.1 Query matching results

Table 2 shows the test results of different methods. Since accuracies are high, we also report cross entropy for a more meaningful comparison. Using the word embeddings of labels already yields high results. However, when combined with aligned graph embeddings with the *Weighted* method the results are significantly better.

4.2 Intrinsic Evaluation

We also evaluate our meta-embeddings intrinsically with a standard word similarity prediction evaluation : we use word embeddings to predict cosine similarity between word pairs, and measure the pearson correlation between cosine similarity and human judgments from similarity/relatedness prediction datasets. SimLex [HRK15] is a similarity judgement dataset (antonyms should have a low rating) while MEN [BTB14] and MTurk [RAGM11] are relatedness dataset (antonyms can have a high rating).

	SimLex	MEN	MTurk
Word2Vec	51.8	81.7	73.3
WordNet2Vec	52.4	39.8	36.2
CONC	**53.6**	81.7	73.3
Best Weighted (RDF2Vec)	51.8	81.7	73.6
Best Weighted (WordNet2Vec)	**53.6**	**82.1**	**74.9**

TABLE 3 – Pearson correlation between cosine similarity of embeddings and human judgments for several models. We used the best values of α when reporting the score of Weighted models. CONC is a meta-ensembling baseline (concatenation of emebeddings).

Once again, we use the Weighted meta-embedding model from equation 2. We report the improvement over the Word2Vec baseline according the the value of of α. Figure 5 shows the results over various datasets. We also performed the same experiment using WordNet2Vec [BAK+17] instead of RDF2Vec. WordNet2Vec is a graph embedding computed using the Wordnet graph, consisting $285k$ relations between words, such as `(furniture, is_a, piece_of_furniture)`

We used the same experimental setup but performed 2 epochs when optimizing $\mathcal{L}$. The results of best Weighted models are reported in table ??. Our meta-embeddings are competitive with CONC while having lower dimensionality (300 vs 1150).

5 Related Work

Several models exist for meta-embedding [YS16] [MSL17]. However, they use a set of embeddings and a return a meta-embedding lying in a new latent space, except [CB18] who shows that meta-embeddings can be obtained by simply averaging or concatenating a set of input embeddings.

Retrofitting models [FDJ+15,] also improve embeddings by leveraging knowledge graphs, in a different way : they use pre-computed word embeddings and tune word representations so that they fulfill some constraints dictated by the knowledge graph.

The most similar approach to ours is[MLS13] where embeddings in different languages (e.g. french and english) are aligned using a translation matrix learn on a limited size multilingual lexicon.

The specificity of our best model is that it is additive. With proper cross validation, the weighted version of

our method can ensure better or equal results.

Another line of work deals with Alignment of knowledge from textual data and graph data. It has been explored with joint learning of embeddings from language model and knowledge graph link prediction [ABMiK18]. However, those methods are less flexible, and ca not leverage the high quality word embeddings trained on massive textual datasets without a re-training from scratch.

6 Conclusion

We proposed a simple, flexible meta-embedding method based on word embeddings and labelled graph embeddings and reported significant improvement on word representation and SPARQL queries/natural language matching. It can be applied to other graphs such as UMLS [BKF+18] for biomedical NLP or social networks graphs [LK14]. Other languages can be used as well. We expect more substantial gains on low resource languages where corpus sizes are more limited.

Acknowledgments

This work has been supported by ERA-Net CHIST-ERA LIHLITH Project funded by the Agencia Estatal de Investigación (AEI, Spain) projects PCIN-2017-118/AEI and PCIN-2017-085/AEI, the Agence Nationale pour la Recherche (ANR, France) projects ANR-17-CHR2-0001-03 and ANR-17-CHR2-0001-04, and the Swiss National Science Foundation (SNF, Switzerland) project 20CH21 174237.

Références

[ABMiK18] Mohammed Alsuhaibani, Danushka Bollegala, Takanori Maehara, and Ken ichi Kawarabayashi. Jointly learning word embeddings using a corpus and a knowledge base. In *PloS one*, 2018.

[BAK+17] Roman Bartusiak, Lukasz Augustyniak, Tomasz Kajdanowicz, Przemyslaw Kazienko, and Maciej Piasecki. Wordnet2vec : Corpora agnostic word vectorization method. *Neurocomputing*, 326-327 :141 150, 2017.

[BKF+18] Andrew L. Beam, Benjamin Kompa, Inbar Fried, Nathan P. Palmer, Xu Shi, Tianxi Cai, and Isaac S. Kohane. Clinical concept embeddings learned from massive sources of medical data. *CoRR*, abs/1804.01486, 2018.

[BSA18] Ander Barrena, Aitor Soroa, and Eneko Agirre. Learning text representations for {500k} classification tasks on named entity disambiguation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 171 180. Association for Computational Linguistics, 2018.

[BTB14] Elia Bruni, Nam-Khanh Tran, and Marco Baroni. Multimodal distributional semantics. *J. Artif. Intell. Res.*, 49 :1–47, 2014.

[CB18] Joshua Coates and Danushka Bollegala. Frustratingly easy meta-embedding – computing meta-embeddings by averaging source word embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies, Volume 2 (Short Papers)*, pages 194 198. Association for Computational Linguistics, 2018.

[FDJ+15] Manaal Faruqui, Jesse Dodge, Sujay Kumar Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 1606–1615. Association for Computational Linguistics, 2015.

[FEMR15] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web Journal, July*, 2015.

[HRK15] Felix Hill, Roi Reichart, and Anna Korhonen. Simlex-999 : Evaluating semantic models with genuine similarity estimation. *Comput. Linguist.*, 41(4) :665–695, December 2015.

[KB15] Diederik P. Kingma and Jimmy Ba. Adam : A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

[LK14] Jure Leskovec and Andrej Krevl. SNAP Datasets : Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[MGB+18] Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

[MLS13] Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168, 2013.

[MSL17] Avo Muromägi, Kairit Sirts, and Sven Laur. Linear ensembles of word embedding models. In *NODALIDA*, pages 96–104. Association for Computational Linguistics, 2017.

[PHH13] Camille Pradel, Ollivier Haemmerlé, and Nathalie Hernandez. Swip, a semantic web interface using patterns (demo) (short paper). In *International Semantic Web Conference (ISWC 2013), Sydney, Australia, 21/10/13-25/10/13*, page (electronic medium), http ://CEUR-WS.org, 2013. CEUR-WS : Workshop proceedings.

[RAGM11] Kira Radinsky, Eugene Agichtein, Evgeniy Gabrilovich, and Shaul Markovitch. A word at a time : Computing word relatedness using temporal semantic analysis. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 337 346, New York, NY, USA, 2011. ACM.

[RP16] Petar Ristoski and Heiko Paulheim. Rdf2vec : RDF graph embeddings for data mining. In *International Semantic Web Conference (1)*, volume 9981 of *Lecture Notes in Computer Science*, pages 498–514, 2016.

[SWW+18] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Chunyuan Li, Ricardo Henao, and Lawrence Carin. Baseline needs more love : On simple word-embedding-based models and associated pooling mechanisms. In *ACL*, 2018.

[TMDL17] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. Lc-quad : A corpus for complex question answering over knowledge graphs. In *International Semantic Web Conference*, 2017.

[YS16] Wenpeng Yin and Hinrich Schütze. Learning word meta-embeddings. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 1351–1360. Association for Computational Linguistics, 2016.