Joint Learning of Entity and Type Embeddings for Analogical Reasoning with Entities

Federico Bianchi, Matteo Palmonari

University of Milano - Bicocca, Viale Sarca 336, Milan, Italy
federico.bianchi@disco.unimib.it, matteo.palmonari@disco.unimib.it

Abstract. Two situations can be considered analogous if they share a common pattern. Analogical reasoning is the task of finding analogies and inferencing missing terms in them. Since natural language is ambiguous, as the same word can refer to different entities, the use of disambiguated entities from Knowledge Graphs for analogical reasoning might bring to better results. Also, entities have types, i.e. classes, in an ontology, from which they inherit characteristics and properties. In this work we focus on a method to represent entities and their types in a joint vector space to do analogical reasoning. We experiment our representations on a dataset that contains analogies on entities and we show that extending the entity representations with information coming from the types improves analogical reasoning results.

1 Introduction

Two situations are analogous if they share a common pattern of relationships among their constituent elements, even though the elements themselves differ across the two situations [6]. An example of analogy is the following: *Rome* is in relation with *Italy* in the same way that *Paris* is in relation with *France*. Analogical reasoning is well-know in natural language and it has been defined has *a kind of reasoning that applies between specific exemplars, in which what is known about one exemplar is used to infer new information about another exemplar* [5]. One simple form of analogy is thus represented using a four-term or *propositional* structure: $a : b :: c : d$; the analogical reasoning task is to infer an unknown term, for example d, that is related to c in the same way that b is related to a [6]; considering the aforementioned example, this would be expressed as the task of inferring that France is related to the term Pairs with the prior knowledge that we know that Rome is related to Italy.

Natural Language Processing (NLP) techniques for generating vector representations of words starting from texts have become popular. For example, word2vec is a model that uses a neural network with one hidden layer to learn word embeddings starting from text [12]. This model is able to learn word representations in a lower dimensional space than the one represented by the one-hot encoding of each word. One interesting property of models like word2vec is that they are able to maintain some of the linguistic regularities of language in the vector space, and thus, analogical reasoning is often possible with

these representations: the vector of the word "Paris" subtracted by the vector of the word "France" and incremented by the vector of the word "Italy", gives in [12] a point in space near the vector that represents the word Rome, $v(\text{“}Paris\text{”}) - v(\text{“}France\text{”}) + v(\text{“}Italy\text{”}) \approx v(\text{“}Rome\text{”})$.

In recent years Knowledge Graphs (KGs) have been used for efficiently modeling information and knowledge. KGs are used to represent real-world entities and the relations between them. Entities in a KG are usually assigned types, i.e. classes defined in an ontology; types contain valuable information that is inherited by the entities. For example, Italy in a KG is assigned the type *country*, thus inheriting all the properties of being a country. Since Named Entity Linking (NEL) techniques [15] can bridge the gap between KGs and text, by finding entity in the text, analogical reasoning is a task that can be tackled with the use of disambiguated entities.

The main difference between analogical reasoning with entities and analogical reasoning with words is that word can be ambiguous: the same word could refer to different entities. As an example, the word "Paris" is commonly associated with the city in France, but there are other cities with the same name, like Paris in Texas.

Also, it has been stated that the analogical reasoning requires some constraints to be placed upon what the analogical relation might be [7]. We can use the type representations to enrich the representation of the entities. This is crucial: while the representation of the entities is generated using only the information given by each entity, the representation of types contain generalized information about all the entities they represent. In the example listed above both Rome and Paris are of type *city* while Italy and France are of type *country*.

In this work we make a few steps towards the use of a joint embedding for entities and types for the task of analogical reasoning with entities. We thus propose a model to represent entities with their own type in a vector space and we evaluate this representation using a dataset that contains analogies with entities.

The main contribution of this work is are: 1) the definition of a model to represent entity and type in the vector space for the task of analogical reasoning. 2) a method to generate the joint representation of the entity and type representations. 3) an experiment on a dataset that contains analogies.

2 Typed Entity Representation and Analogical Reasoning

Our main hypothesis is that a joint representation of entities and their own type can bring improvements in the task of analogical reasoning with entities. To do this we have to generate a vector representation for both entities and types. We will thus refer to Entity Representation (ER) as the representation of entities in an n-dimensional space, while Type Representation (TR) will be used to refer to the representation of the types in m-dimensional space. ER and TR can be connected in the same fashion as [11] where for each entity in ER we concatenate its own vector with the vector of the respective type

found in TR, thus obtaining a Typed Entity Representation (TER), which is the representation of typed entities in a (n+m)-dimensional vector space. The TER model is thus built upon existing ER and TR. The TER model has three main advantages: 1) In the enriched vector space represented by TER the entities that have the same type are now closer to each other, since they all share the same type vector to represent them. 2) With this representation we could also answer to analogies like "Which is the Book that correspond to the Movie The Matrix?" by subtracting the type vector representing "Film" and adding the vector of the type "Book" to the vector of the entity "The_Matrix", in TER, thus combining operation on types and operations on entities. 3) Since in TER entities are now extended with their own type, analogical reasoning can be done with done with the support of this enriched space.

Our representations are learned on short text descriptions for each entity that can be found inside the DBpedia abstracts dataset (we use 2015-10 dataset). In the following, we describe the process that we used to represent the entities in a vector space with the use of word2vec:

1. we retrieve the DBpedia abstract for each entity and we annotate it using the DBpedia Spotlight[1] annotator, bridging the gap between the text and the DBpedia KG. After these steps we obtain a set of entities extracted from the input text
2. we remove all the words that were not annotated and we leave only the entities inside the text corpus
3. the text is fed to the word2vec model that is going to learn the ER model

DBpedia provides also an ontology: entities in the KG are characterized by a series of types (e.g. Berlin has types Settlement, Location, City...) and we want to learn a vector representation for those too. To learn vector representations for the types we use the same procedure listed above with just one variation: instead of feeding word2vec with documents containing the entities, we replace, for each entity, its own type. We consider the single type defined in the DBpedia-2015 instance type dataset[2] to replace each entity with its own type.

In Figure 1 we show an example of annotation for both entities and types that are then given in input to the word2vec model, the example text has been taken from the DBpedia Spotlight demo, while the types are again defined using the DBpedia-2015 instance type dataset.

Word2vec will thus learn the representation of types in the same way it learned the representation of the entities. The model we obtained allows us to define analogical reasoning operations with types, for example $v(dbo:Book) - v(dbo:Writer) + v(dbo:Film) \approx v(dbo:ScreenWriter)$, where *dbo* is used to identify the ontology namespace in DBpedia (http://dbpedia.org/ontology/). The final step to generate the joint embedding is thus the concatenation of the vectors, in Figure 2 we show an example of the concatenation of an entity vector (of entity Barack Obama), coming from ER, with its own type vector (Office Holder), coming from TR.

[1] `http://demo.dbpedia-spotlight.org/`

[2] `http://wiki.dbpedia.org/dbpedia-dataset-version-2015-10`

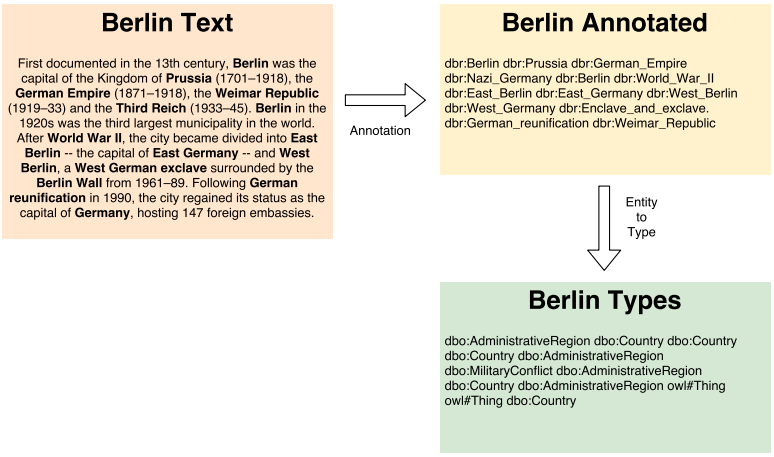

Fig. 1. An example of the annotation of text to generate the input corpus for the word2vec models

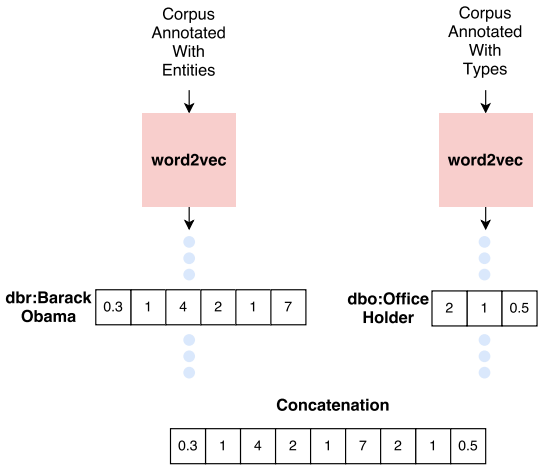

Fig. 2. An example of concatenation between an entity and its own type

A few other examples about analogies obtained by the TER model are visible in Table 1 in which we also show a possible explanation on why we got that result (some entities' names have been shortened). With the use of disambiguation, the word Paris is now not ambiguous in the dataset and we can use it for different analogies: in the Table we show analogies with both Paris, in France, and Paris in Texas.

Table 1. Examples of TER results with possible explanations

Operation	Result	Possible Explanation
v("Demi_Lovato") - v("United_States") + v("Germany")	v("Stefan_Raab")	Stefan Raab (German), singer as Demi Lovato (American)
v("Michael_Bradley_(soc.)") - v("United_States") + v("France")	v("Marcel_Desailly")	Bradley (American), soccer player as Desailly (French)
v("Barack_Obama") - v("Democr._Par.") + v("Repub._Par.")	v("John_McCain")	Democratic and Republican candidate for US election
v("Paris") - v("France") + v("Germany")	v("Berlin")	Both Capitals, one in France and one in Germany
v("Pairs,_Texas") - v("United_States") + v("Germany")	v("Ueckermunde")	Both Cities, one in the US and one in Germany
v("Barack_Obama") - v("God") + v("Satan")	v("Richard_Nixon")	Nixon has a bad reputation

3 Experiments

Motivation In this section we are going to test the performance of TER with respect to ER for the analogical reasoning task with entities. We will then analyze the errors that both representation make during the evaluation.

Dataset We considered the analogy dataset[3] where other methods in the state-of-the art have been experimented on for analogical reasoning with words. There are two main things to notice with this dataset and that required edits on our side:

- the dataset contains both semantic analogies and syntactic analogies, but with the use of annotation in our approach we lose the ability to test syntactic analogies, since only entities are extracted. We thus tested only semantic analogies.
- one of the main advantages of using NEL techniques to detect entities, is that there are less ambiguities about the meaning of words (for example Georgia, in the U.S., is different from Georgia, in Europe); since disambiguation techniques like the ones provided by spotlight change the name of the entity to disambiguate them (Georgia_(U.S._State) and Georgia_(Country), in the aforementioned example), we annotated the word inside the analogy dataset and replaced them with their own DBpedia URI.

The dataset contains pair of relations between cities and their own state. A few examples extracted from the dataset can be seen in Table 2, where we show both the original pairs and the one we replaced, we use the *dbr* prefix to identify the DBpedia URI for resources (http://dbpedia.org/resource/). This dataset was manually annotated: some of the annotations where straightforward, since some of the capitals and states can be mapped directly to their own DBpedia page (e.g. Athens and dbr:Athens). This is not true for cities in the United States which required more attention in the annotation (e.g. Oakland and dbr:Oakland,_California). The analogical reasoning task is: given the first three elements, find the fourth one. We would thus like the nearest point in space at $v(SecondElement) - v(FirstElement) + v(ThirdElement)$ to be

[3] `https://aclweb.org/aclwiki/Google_analogy_test_set_(State_of_the_art)`

$v(FourthElement)$. We evaluate the model using accuracy: the model gives the correct answer to a vector operation like the one presented above when the nearest point in the vector space, after the vector operation, is the FouthElement.

Table 2. Table that shows a sample of the dataset we used for evaluation, in the parenthesis we show the annotated version of each word element

Word (DBpedia Entity)			
First	**Second**	**Third**	**Fourth**
Athens (dbr:Athens)	Greece (dbr:Greece)	Beijing (dbr:Beijing)	China (dbr:China)
Portland (dbr:Portland,_Oregon)	Oregon (dbr:Oregon)	Oakland (dbr:Oakland,_California)	California (dbr:California)
Worcester (dbr:Worcester,_Massachusetts)	Massachusetts (dbr:Massachusetts)	Atlanta (dbr:Atlanta)	Georgia (dbr:Georgia_(U.S._state))
Luanda (dbr:Luanda)	Angola (dbr:Angola)	Tbilisi (dbr:Tbilisi)	Georgia (dbr:Georgia_(country))

Algorithms and Baseline We will test the performance of different configurations of the proposed approach using the *Continuous Bag of Words* Model (CBOW). As explained before, the main parameters that will change are the features size and the dimension of the windows to consider. We will also test a baseline to compare our approach to the standard models. As our baseline we decide to use a CBOW model on the non annotated corpus. Performance will be evaluated on the non annotated version of the aforementioned dataset. While the dataset is different, and thus the comparison might not be considered fair, this baseline will help to shed light on the differences between the annotated and the non annotated version of this analogical task. We also remove standard English stopwords and punctuation from the corpus.

Replication of the Experiments To replicate our experiments we provide the configuration of our algorithms in the next section. We also provide the code[4] to run our algorithms and the annotated DBpedia abstracts, that can be downloaded and easily used in the model. For our experiment we considered the Continuous Bag of Words model, as defined in [12]. The model also removed from the corpus those words that appeared less then 5 times (min-count parameter). There are two main hyper-parameters in the model: the first one is the window size: since the model learns representations from text, it has to know how many words to consider around each word for which we want to learn the representation; the second parameter is the number of features (or dimensions) that will be used to represent the word. Further details about the model are presented in [12].

Results One of the entity of the gold standard, dbr:Tallassee,_Tennessee, was not found inside our vector space. This might be related to the fact that this entity is not really common inside Wikipedia and that it might have been removed in the annotation phase. All the pairs (78) containing this entity were excluded from the evaluation, leaving us with a dataset of 7420 pairs.

The following models where run multiple times since there are slight variations in the position of the vectors in the space at each run of the algorithm. We here report the results based on the average of 5 different runs.

[4] `https://github.com/vinid/entity2vec`

We first tested the ER model, to define a first baseline. We tested different configurations of the model and we realized that when the number of the features or the length of the window is too high or too low, the performance decreases; we thus reported only the most meaningful results, that are visible in Table 3. The best model was obtained with the use of a window of length 3 and 100 features. We generated different parameterization for TR and then we concatenated with ER to obtain and evaluate the accuracy of TER. We report only the result of the TER built on ER(window = 3, features = 100) with different TR configurations, that had the best performance, in Table 4.

Table 3. Results of the ER model with different parameters

Window	Features	Accuracy
2	100	0.78
2	200	0.78
3	100	**0.79**
3	200	0.78
5	100	0.77
5	200	0.75

Table 4. Results of the TER model with different parameters built upon an ER(window = 3, features = 100)

Window	Features	Accuracy
2	25	**0.86**
2	50	0.84
2	100	0.84
2	200	0.84
3	25	0.84
3	50	0.82
3	100	0.82
3	200	0.82
5	25	0.83
5	50	0.76
5	100	0.77
5	200	0.78

We can see that TER is able to improve the performance in the analogical reasoning task with respect to ER. The best result was obtained by TER with ER(window = 3, features = 100) and TR(window = 2, features = 25), and thus by extending the 100 dimensional vector of each entity with 25 dimension coming from its own type. The performance decreases as soon as the window or the number of feature is extended too much. Also, we show in Table 5 the comparison of the best ER model, the best TER model and the best result achieved with the use of the baseline.

Table 5. Comparison of algorithms and baseline

Model	Accuracy
TER(window=2, features=25)	**0.86**
ER (window=3, features=100)	0.79
Baseline (window=10, features=400)	0.63

The baseline algorithm on the non annotated dataset has lower accuracy then the ER and TER models, this could be the result of two main factors: 1) Baseline has to represent a bigger amount of data (this also explains why the number of features in the model is 400); 2) The annotation phase of the dataset helps to remove some ambiguities on the analogical reasoning task (e.g. Georgia).

Error Analysis Since there is a difference in the performance of the two models we analyzed the errors of the models that performed better: 1) TER built concatenating TR(window = 2 features = 25) with ER(window = 3, features = 100) 2) ER(window = 3, features = 100), to see which were the main differences. Errors are of mainly three types: those errors made by ER, those errors made by TER and those errors that are made by both ER and TER. A small sample of these errors can be seen in Table 6, where we also show an example of analogy that was identified by both models. One interesting result that we can point out from this small table is that the use of types seems able to correct some of the errors that ER does, for example sometimes, where the answer should be United Kingdom, the answer of ER is Great Britain. This is not completely wrong, since the analogy can be considered still valid at a certain degree, but the answer United Kingdom seems more appropriate, since China and United Kingdom are both Country, while Great Britain is an Island. Types, enriching the entities vectors, are able to enforce coherence in the analogical reasoning task with entities.

Table 6. Examples of the output entities with different pairs of analogies

Kind	First	Second	Third	Correct Answer	TER	ER
Both Model Ok	Hanoi	Vietnam	London	United_Kingdom	**United_Kingdom**	**United_Kingdom**
ER Wrong	Beijing	China	London	United_Kingdom	**United_Kingdom**	Great_Britain
ER Wrong	Berlin	Germany	Canberra	Australia	**Australia**	Australian_Capital_Territory
ER Wrong	Bern	Switzerland	Tehran	Iran	**Iran**	Politics_of_Iran
ER Wrong	Cairo	Egypt	London	United_Kingdom	**United_Kingdom**	Great_Britain
TER Wrong	Helsinki	Finland	Berlin	Germany	Saxony	**Germany**
Both Model Wrong	London	United_Kingdom	Moscow	Russia	Soviet_Union	Soviet_Union

In the dataset pairs are divided in mainly two groups: the first one contains general capital-nation analogies, while the second contains city-state analogies about the United States. We thus analyzed the performance of the models evaluating the pairs from the first to the last: the results can be seen in Figure 3. This Figure shows the accuracy, in term of correct analogies with respect the total number of analogies in the dataset; the x-axis shows the number of pairs evaluated incrementally. We can mainly see two drops in performance in both models, the first one happens in the first 1000 pairs of analogies and it is mostly due to error generated by those entities that might be less mentioned inside the abstracts (e.g. Samoa, Jordan and Tajikistan). Also, this drop is due to the low number of analogies evaluated until that moment: a few more errors have a huge impact on the first accuracy results. The second drop in performance happens

after the 5000th pairs and is related to the pairs that contains cities and states from the United States. This last drop seems to be a bit worse for the ER model: we counted the errors made by both model in the last part of the dataset (those part that contains analogies based on cities and states in the United States). TER was wrong 32 times, where ER was right, while ER was wrong 337 times in pairs where TER got the answers right. We can say that adding types has reduced the number of errors for those analogies with United States cities/states.

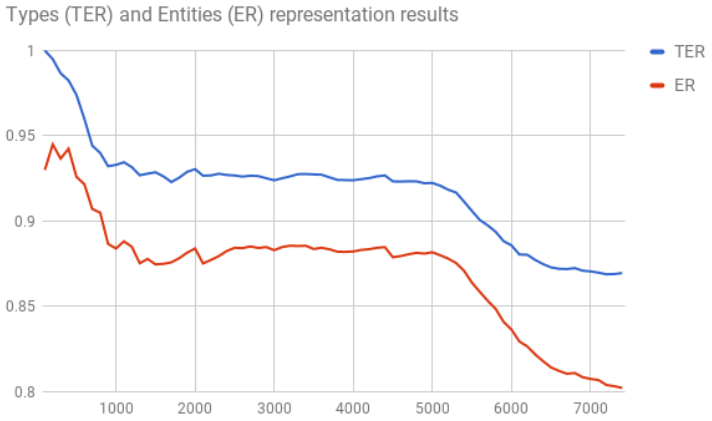

Fig. 3. ER and TER results with pairs on the x-axis

4 Related Work

Analogy has been already studied in literature in the context of KGs. For example, in [16], a framework for doing deductive and analogical reasoning on a KG is defined. The method is based on vector representation to do reasoning in the following way: if we want to find the connection between Micheal Jackson and music, we could look at two KG triples: *(MichealJackson, is a, songwriter)* and *(musician, composes, music)* even if there is no strict chain of reasoning we could find a path that connects the two facts by the fact that songwriter and musician are near in space. Experiments were done to evaluate the performance of the solver used for finding paths, while in our case we wanted to directly evaluate the performance on the analogical reasoning task with entities. Analogy as also been used in [9] for the task of knowledge base integration, where the analogy is mainly used for mappings between different domains and not inside the same domain like in the case of our work. A less recent work uses dimensionality reduction to find analogies in knowledge bases [8].

Models to represent entities in a vector space have already been defined in literature. There are some works related to the representation of entities using

word2vec models. One example of these is presented in [1] where a learning to rank approach is used to evaluate entity relatedness using different features. One of the features considered is obtained using word2vec on documents, extracted from Wikipedia, composed by only the entities appearing inside the document. The main difference with our approach related to the generation of the entity vectors is that we detect entities using an annotator, while they detect entities considering the manual link that the editor of the page has inserted in the text. Our approach can find more entities, since some of the links might be missing in the text. Also, in our approach the same entity could be found multiple times, since when an entity is repeated in the text, the editor usually does not insert the same link over and over: if the entity is mentioned with others entities at the end of the text, it might be interesting to have it annotated again to gain new information. On the other side we rely on an annotator that could make errors in the annotation phase.

In [18] a multi-level representation is used to predict the probability of an entity e of having type t. Representation are given at three different levels, character-level, word-level, and entity-level; experiments show that the joint representation of the three levels performs better than each level used alone.

Embeddings for entities have been also studied in different contexts, for example in [14] a tensor factorization model is presented and tested on classification and link prediction tasks. The knowledge graph is originally represented as a 3-way order tensor, where each dimension is associated with the subject entity, the relation and object entity of a triple, respectively. An extension of this work uses types to improve performance of the model and is defined in [3]. A different approach for entity and relation embedding has been tackled in different works [10] [17] [2] where entities and relations are projected in a vector space. In [2] the main idea is that if we start from a triple (h, l, t), the embedding of the entity h plus a vector that depends on the relationship l should give as result a point in the space that is close to the embedding of the entity t. In [10] the work is extended allowing the possibility of easily representing N-to-N relationships between entities by representing data using a projection matrix that projects entities from their own space into the relation space.

Another interesting approach, defined in the context of word embedding, is presented in [4], where semantic lexicons like WordNet [13] are used to improve the word vectors: in their work they refine the vector space in a way that linked words in the semantic lexicon have similar vector representations. Also, in [19], semantic knowledge is again used to improve word embeddings.

5 Conclusion and Future Work

Conclusion In this paper we have presented a method to tackle analogical reasoning with entities. To do this we represent entities and types in a vector space and we used those two representations to generate a combined representation. To evaluate our work we modified one of the datasets used for the analogical reasoning task with words by annotating it with entities. Experimental results

have shown that the use of the joint representation of entities and types can improve the results with respect to the representation of entities alone. We found that both our models benefit from a small window, and this can be mainly due to the removal of those words that were not annotated by the annotator. This removal has both an advantage and a disadvantage: while we are able to remove a lot of words and to pass to the word2vec model only those entities that are found, we lose the ability to represent other elements, like verbs. Also, some of the answers got by ER are not completely wrong: if we consider Table 6 we could consider correct the answer Great Britain, because in common language Great Britain can be used to identify the United Kingdom country.

Future Work The work presented in this paper can be improved in many ways, in the following we list some of the thing we might consider for moving forward:

- the choice of the corpus: we used the DBpedia abstracts, but these often contain general information. It would be interesting to apply this methods to the whole Wikipedia content
- the annotation tool: we used the DBpedia Spotlight endpoint that takes parameters in input. Changing these parameters can bring to huge differences in the annotated output. A study on the performance of the algorithms using different parameterizations might be conducted. We could also use a different annotation tool like TextRazor[5] or Dandelion[6], since sometimes Spotlight makes errors in the annotation (in Figure 1 the entity dbr:Enclave_and_exclave might not be considered correct
- type representation evaluation: we were not able to directly test our representation of types (TR), since we were not able to find a dataset to evaluate the similarity of these concepts. One of our next steps is to collect data to perform an evaluation
- semantic similarity: this kind of representations can be used to evaluate the similarity between entities. TER might give different insights on similarity: the entity Mark Zuckerberg could be more similar to Steve Jobs than it is to Facebook, since the first two are of type Person (and also share some other characteristics) while the last one is of type Company
- predicate representation: one of the main issues with our model is that it is not able to represent the relation between the entities, the next steps in this case would be to also represent the predicate in the space starting from text

6 Acknowledgement

We thank Pierpaolo Basile for his valuable suggestions on how to approach this problem.

[5] `https://www.textrazor.com/`
[6] `https://dandelion.eu/`

References

1. Basile, P., Caputo, A., Rossiello, G., Semeraro, G.: Learning to rank entity relatedness through embedding-based features. In: NLDB. pp. 471–477. Springer (2016)
2. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: Advances in neural information processing systems. pp. 2787–2795 (2013)
3. Chang, K.W., Yih, S.W.t., Yang, B., Meek, C.: Typed tensor decomposition of knowledge bases for relation extraction. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. ACL Association for Computational Linguistics (October 2014), https://www.microsoft.com/en-us/research/publication/typed-tensor-decomposition-of-knowledge-bases-for-relation-extraction/
4. Faruqui, M., Dodge, J., Jauhar, S.K., Dyer, C., Hovy, E., Smith, N.A.: Retrofitting word vectors to semantic lexicons. In: Proceedings of NAACL (2015)
5. Gentner, D.: Analogical reasoning, psychology of. Encyclopedia of cognitive science (2003)
6. Holyoak, K.J.: Analogy and relational reasoning. The Oxford handbook of thinking and reasoning pp. 234–259 (2012)
7. Holyoak, K.J., Thagard, P.: Analogical mapping by constraint satisfaction. Cognitive science 13(3), 295–355 (1989)
8. Krishnamurthy, J.: Finding analogies in semantic networks using the singular value decomposition. Ph.D. thesis, Massachusetts Institute of Technology (2009)
9. Kuo, Y.L., Hsu, J.Y.j.: Bridging common sense knowledge bases with analogy by graph similarity. In: Collaboratively-Built Knowledge Sources and AI (2010)
10. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI. pp. 2181–2187 (2015)
11. Liu, Y., Liu, Z., Chua, T.S., Sun, M.: Topical word embeddings. In: AAAI. pp. 2418–2424 (2015)
12. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems. pp. 3111–3119 (2013)
13. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM 38(11), 39–41 (1995)
14. Nickel, M., Tresp, V., Kriegel, H.P.: A three-way model for collective learning on multi-relational data. In: Proceedings of ICML-11. pp. 809–816 (2011)
15. Rizzo, G., Troncy, R.: Nerd: a framework for unifying named entity recognition and disambiguation extraction tools. In: Proceedings of EACL. pp. 73–76. Association for Computational Linguistics (2012)
16. Summers-Stay, D.: Deductive and analogical reasoning on a semantically embedded knowledge graph. arXiv preprint arXiv:1707.03232 (2017)
17. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI. pp. 1112–1119 (2014)
18. Yaghoobzadeh, Y., Schütze, H.: Multi-level representations for fine-grained typing of knowledge base entities. In: Proceedings of EACL 2017. pp. 578–589. Association for Computational Linguistics (2017)
19. Yu, M., Dredze, M.: Improving lexical embeddings with semantic knowledge. In: ACL (2). pp. 545–550 (2014)