Make Embeddings Semantic Again!

Heiko Paulheim

Data and Web Science Group, University of Mannheim, Germany
heiko@informatik.uni-mannheim.de

Abstract. The original Semantic Web vision foresees to describe entities in a way that the meaning can be interpreted both by machines and humans. Following that idea, large-scale knowledge graphs capturing a significant portion of knowledge have been developed. In the recent past, vector space embeddings of semantic web knowledge graphs – i.e., projections of a knowledge graph into a lower-dimensional, numerical feature space (a.k.a. *latent feature space*) – have been shown to yield superior performance in many tasks, including relation prediction, recommender systems, or the enrichment of predictive data mining tasks. At the same time, those projections describe an entity as a numerical vector, without any semantics attached to the dimensions. Thus, embeddings are as far from the original Semantic Web vision as can be. As a consequence, the results achieved with embeddings – as impressive as they are in terms of quantitative performance – are most often not interpretable, and it is hard to obtain a justification for a prediction, e.g., an explanation why an item has been suggested by a recommender system. In this paper, we make a claim for *semantic embeddings* and discuss possible ideas towards their construction.

Keywords: Knowledge Graphs, Embeddings, Representation Learning

1 Introduction

Knowledge Graphs are directed, labeled graphs that encode knowledge about entities from multiple domains [7]. *Embeddings* form a projection of those knowledge graphs into a lower dimensional, so-called *latent* or *embedding* vector space, where each entity is represented as a point in that space. Such an embedding space has the following key features:

Density the individual dimensions are not sparse (i.e., the do not contain mainly zeros), but exhibit an even distribution of values. This way, the information content of each dimension is maximized.

Clustering similar entities have a similar position in the vector space.

Relation preservation relations between entities are represented by a constant vector representation. That way, (approximate) arithmetics such as $\overrightarrow{Germany} + \overrightarrow{capital} \approx \overrightarrow{Berlin}$ become possible.

Fig. 1 depicts selected entities of DBpedia and Wikidata in the two dimensional PCA projection of an embedding space and illustrates these properties. Countries and cities form clusters, and the displacement between a country and its capital is similar for all the countries.

Over the past years, quite a few approaches for generating such embeddings have been proposed, including TransE [1], TransH [11], and TransR [6], NTN [10], RDF2vec and its variants [2, 8], RDFGlove [3], and many others. It has been shown that they perform well on tasks such as relation prediction [11], content-based recommender systems [9], as well as prediction problems with background knowledge in RDF datasets [8].

While the quantitative performance on those tasks is undisputed, the semantics of the knowledge representation are given up in favor of an entirely numeric, non-semantic representation. Therefore, the use of embeddings does not allow the interpretation of results. However, interpretations are useful, e.g., for

- justifying *why* a relation was added to a knowledge graph,
- explaining *why* an item was suggested by a recommendation system, or
- allowing for *descriptive*, not only *predictive* data mining.

With embedding methods such as the ones enumerated above, the only explanations a user can get look like *The item is close in the latent vector space to other items you liked*, or *Resources which have a value larger than 0.412 on dimension 108 usually have the property*. Compared to the original Semantic Web vision, which foresees maximal interpretability of data and justifications, these methods exist at the opposite end of the design space, i.e., they are as non-semantic as can be. Hence, we argue for the need of *semantic* embeddings, which preserve the interpretability.

2 Towards Semantic Embeddings

As a *semantic* embedding, we understand an embedding space where each dimension can be assigned a human interpretable meaning. For example, for cities, such dimensions could be *size* or *economic wealth*. In terms of the underlying knowledge graph, they can map to elementary properties (e.g., *population*) or complex constructs (e.g., number of *headquarter* relations from companies).

2.1 A Posteriori Learning of Interpretations

One possible approach to solve this issue could be to learn a semantic interpretation for each dimension *a posteriori*, i.e., training a symbolic regression model to predict the value of each dimension based on interpretable features extracted from the knowledge graph. In theory, this works on any possible embedding, but there is no guarantee that such an approach would actually yield a meaningful model, and, depending on the task and type of model, it would still not be a trivial task to exploit the model for coming up with a semantic explanation.

2.2 Pattern-based Embeddings

Another possible approach is deriving embedding dimensions from patterns observed in a knowledge graph. While the design space for embedding construction is arbitrarily large – i.e., each dimension can be any arbitrary function – pattern-based embeddings would work differently. They guide the process of computing an embedding by restricting that search space so that each dimension corresponds to a pattern in the data, or combinations thereof. This allows tracking back the origin of a dimension to the original pattern(s) from which it was derived, and thereby fosters the interpretation of embedding dimensions.

We propose to first learn a set of universal patterns that hold for a knowledge graph, e.g., by learning graph patterns [4] or Horn clauses [5]. As a first step towards a semantic embedding, each of those patterns can be regarded as a latent feature – either binary (i.e., a resource exposes the pattern or not) or continuous (i.e., a resource exposes the pattern *to a certain extent*). This would provide a first approximation of a semantic embedding.

However, at this stage, the embedding space might still be high-dimensional and sparse, because it might take a lot of patterns to describe a knowledge graph. Therefore, a second step is required for *compacting* the embedding space. This could be achieved, e.g., by identifying patterns that are completely or almost mutually exclusive, and combining them in compacted dimensions. For example, only movies may have Oscar-winning actors, whereas only cities may be located in Europe (but movies have no location, and cities have no actors). Therefore, we could define an embedding dimension combining those two patterns, exposing different semantics depending on the type of entity at hand.

3 Challenges

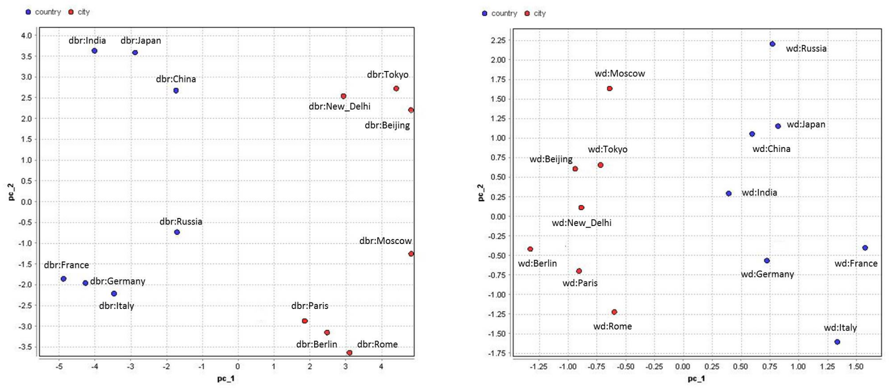
While some works on pattern induction in RDF graphs exist, it is not trivial to assume that each pattern is also a good candidate for an embedding dimension (or a partial one). Therefore, we need to develop heuristics for identifying patterns that are suitable as latent features, and ideally adapt the pattern learners so that they focus on finding such patterns.

Since embedding spaces should be continuous, another challenge is to find patterns that are not just binary (e.g., an instance has a type or not), but continuous in a sense that they can be fulfilled to a certain extent. Here, involving numerical and date-valued literals would also be desired, since they are often neglected in current embedding approaches, but contain a lot of useful semantics.

Finally, the dimensionality reduction of the pattern-based embeddings also needs to be carefully designed. We can assume that compacting will ultimately lead to some information loss, but it may even lead to false information. If we, as in the example above, combine *a lot of* features for movies and cities, this will ultimately render movies and cities to be indistinguishable in the vector space.

4 Conclusion

While embeddings have been proven to yield superior quantitative performance on many tasks, a vector space embedding is a very non-semantic representation of a knowledge graph. Hence, we consider *semantic embeddings* as a research



(a) DBpedia vectors (b) Wikidata vectors

Fig. 1: Two-dimensional PCA projection of RDF2vec vectors of countries and their capital cities for DBpedia and YAGO [8]

direction with a lot of potential. We assume that since semantic embeddings are more constrained than generic embeddings, they will not reach their full quantitative performance. We rather see a continuous design space for knowledge graph embeddings with a trade-off between interpretability and quantitative performance.

References

1. Bordes, A., Weston, J., Collobert, R., Bengio, Y., et al.: Learning structured embeddings of knowledge bases. AAAI 6(1) (2011)
2. Cochez, M., Ristoski, P., Ponzetto, S.P., Paulheim, H.: Biased graph walks for rdf graph embeddings. In: WIMS. p. 21. ACM (2017)
3. Cochez, M., Ristoski, P., Ponzetto, S.P., Paulheim, H.: Global rdf vector space embeddings. In: ISWC. pp. 190–207. Springer (2017)
4. Hees, J., Bauer, R., Folz, J., Borth, D., Dengel, A.: An evolutionary algorithm to learn sparql queries for source-target-pairs. In: EKAW. pp. 337–352 (2016)
5. L.A. Galárraga et al.: Amie: association rule mining under incomplete evidence in ontological knowledge bases. In: WWW. pp. 413–422
6. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI. vol. 15, pp. 2181–2187 (2015)
7. Paulheim, H.: Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic web 8(3), 489–508 (2017)
8. Ristoski, P., Paulheim, H.: Rdf2vec: Rdf graph embeddings for data mining. In: ISWC. Springer (2016)
9. Ristoski, P., Rosati, J., Noia, T.D., Leone, R.D., Paulheim, H.: Rdf2vec: Rdf graph embeddings and their applications. Semantic Web (to appear)
10. Socher, R., Chen, D., Manning, C.D., Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In: NIPS. pp. 926–934 (2013)
11. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI. vol. 14, pp. 1112–1119 (2014)