An Open-World Extension to Knowledge Graph Completion Models

Haseeb Shah,[1] Johannes Villmow,[2] Adrian Ulges,[2] Ulrich Schwanecke,[2] Faisal Shafait[1]
[1]National University of Science and Technology, Pakistan
[2]RheinMain University of Applied Sciences, Germany
{hshah.bese15seecs, faisal.shafait}@seecs.edu.pk
{johannes.villmow, adrian.ulges, ulrich.schwanecke}@hs-rm.de

Abstract

We present a novel extension to embedding-based knowledge graph completion models which enables them to perform open-world link prediction, i.e. to predict facts for entities unseen in training based on their textual description. Our model combines a regular link prediction model learned from a knowledge graph with word embeddings learned from a textual corpus. After training both independently, we learn a transformation to map the embeddings of an entity's name and description to the graph-based embedding space.

In experiments on several datasets including FB20k, DBPedia50k and our new dataset FB15k-237-OWE, we demonstrate competitive results. Particularly, our approach exploits the full knowledge graph structure even when textual descriptions are scarce, does not require a joint training on graph and text, and can be applied to any embedding-based link prediction model, such as TransE, ComplEx and DistMult.

1 Introduction

Knowledge graphs are a vital source for disambiguation and discovery in various tasks such as question answering (Ferrucci et al. 2010), information extraction (Dong et al. 2014) and search (Singhal 2012). They are, however, known to suffer from data quality issues (Paulheim 2017). Most prominently, since formal knowledge is inherently sparse, relevant facts are often missing from the graph.

To overcome this problem, *knowledge graph completion* (KGC) or *link prediction* strives to enrich existing graphs with new facts. Formally, a knowledge graph $\mathcal{G} \subset E \times R \times E$ consists of facts or triples $(head, rel, tail)$, where E and R denote finite sets of entities and relations respectively. Knowledge graph completion is targeted at assessing the probability of triples not present in the graph. To do so, a common approach involves representing the entities and relations in triples using real-valued vectors called embeddings. The probability of the triple is then inferred by geometric reasoning over the embeddings. Embeddings are usually generated by learning to discriminate real triples from randomly corrupted ones (Nickel et al. 2016; Shi and Weninger 2017b; Trouillon et al. 2016).

A key problem with most existing approaches is that the plausibility of links can be determined for known entities only. For many applications, however, it is of interest to infer knowledge about entities not present in the graph. Imagine answering the question "What is German actress *Julia Lindig* known for?", where *Julia Lindig* is not a known entity. Here, information can be inferred from the question, typically using word embeddings (Mikolov et al. 2013; Pennington, Socher, and Manning 2014; Peters et al. 2018). Similar to entity embeddings, these techniques represent words with embedding vectors. These can be pre-trained on text corpora, thereby capturing word similarity and semantic relations, which may help to predict the plausibility of the triple $(Julia\_Lindig, starred\_in, Lola\_Rennt)$. This challenge is known as open-world (or zero-shot) KGC. To the best of our knowledge, few open-world KGC models have been proposed so far, all of which are full replacements for regular KGC models and require textual descriptions for all entities (Xie et al. 2016; Shi and Weninger 2017a).

In this paper, we suggest a different approach, namely to extend existing KGC models with pre-trained word embeddings. Given an new entity, we aggregate its name and description into a text-based entity representation. We then learn a transformation from text-based embedding space to graph-based embedding space, where we can now apply the graph-based model for predicting links. We show that this simple approach yields competitive results, and offers two key benefits: First, it is independent of the specific KGC model used, which allows us to use multiple different link prediction models from which we can pick the best one. Second, as training on the graph structure happens independently from training on text, our approach can exploit the full-scale knowledge graph structure in situations where textual information is scarce because learning the transformation is robust even for such situations. We coin our approach OWE for Open World Extension and combine it with several common KGC models, obtaining TransE-OWE, DistMult-OWE, and ComplEx-OWE.

We demonstrate competitive results on common datasets for open-world prediction, and also introduce a new dataset called FB15k-237-OWE, which avoids bias towards long textual descriptions and trivial regularities like inverse relations. The code and the new FB15k-237-OWE dataset are available online[1].

[1]https://github.com/haseebs/OWE

2 Related Work

Knowledge Graph Completion Interest in KGC has increased in the past few years, with a focus on embedding-based methods. A concise survey of earlier works such as NTN (Socher et al. 2013) and TransE (Bordes et al. 2013) is provided by (Nickel et al. 2016). TransE has been recently complemented by other models like DistMult (Yang et al. 2014), ComplEx (Trouillon et al. 2016), ProjE (Shi and Weninger 2017b) and RDF2Vec (Ristoski and Paulheim 2016).

A common approach is to estimate the probability of triples $(head, rel, tail)$ using a scoring function $\phi(u_{head}, u_{rel}, u_{tail})$, where u_x denotes the embedding of entity/relation x and is a real-valued or complex-valued vector. ϕ depends on the model and varies from simple translation (Bordes et al. 2013) over bilinear forms (Yang et al. 2014) to complex-valued forms (Trouillon et al. 2016). Training happens by randomly perturbing triples in the graph and learning to discriminate real triples from perturbed ones, typically by using negative sampling.

Word Embeddings Embedding-based representations of text have become a common approach in natural language processing (Goldberg 2016). They represent terms, sentences or documents in form of a vector. Word embeddings are known to capture term similarities and semantic relations (Mikolov et al. 2013; Pennington, Socher, and Manning 2014). Word embeddings can also include subword information for out-of-training-vocabulary generalization (Bojanowski et al. 2017; Peters et al. 2018), and have been combined with anchor and link information obtained from Wikipedia to produce entity-specific embeddings (Yamada et al. 2016).

Text-Enhanced Knowledge Graph Completion While the knowledge graph completion models described above leverage only the triple structure of the graph, some approaches combine text information with the graph information. Some of these approaches regularize word embeddings based on semantic information, for example, by adding synonym and other relations from WordNet to the training set of contextual term pairs (Faruqui et al. 2015; Yu and Dredze 2014) or by modelling relations like synonyms with translations (Xu et al. 2014). These methods, however, are not targeted at KGC.

Closer to our work are actual KGC models that employ text information, such that for entities scarcely linked in the graph, extra information can be drawn from text. (Socher et al. 2013; Wang and Li 2016) use averaged pretrained word vectors (CBOW) as entity descriptions for geometric reasoning with affine mappings. (Xu et al. 2017) tests different aggregation functions for word embeddings (CBOW, LSTMs, attentive LSTMs) and proposes learning an entity-wise linear interpolation between graph-based and text-based embedding, which is combined with a translational KGC model. (Toutanova and Chen 2015) enhance KGC with a large set of relations extracted from dependency parses on a text corpus, and formulate a joint loss function for text-based and graph-based inputs. However, none of these works address the open-world setting that we target in this paper.

Only few other works address open-world KGC. *Description-Embodied Knowledge Representation Learning (DKRL)* (Xie et al. 2016) uses a joint training of graph-based embeddings (TransE) and text-based embeddings while regularizing both types of embeddings to be aligned using an additional loss. ConMask (Shi and Weninger 2017a) is a text-centric approach where text-based embeddings for head, relation and tail are derived by an attention model over names and descriptions, and the triple (h, r, t) is scored by a pairwise matching of the embeddings, followed by a softmax regression. In contrast to these approaches, we train graph and text embeddings independently. This comes with two benefits: First, our approach can fully leverage knowledge graphs with incomplete or scarce textual descriptions. Second, it is applicable to any embedding-based link prediction model such as TransE, DistMult and ComplEx.

3 Approach

Our approach starts with a regular link prediction model (in the following also referred to as the *graph-based* model) as outlined in Section 2 and visualised in Fig. 1. The model scores triples (h, r, t):

$$score(h, r, t) = \phi(u_h, u_r, u_t) \tag{1}$$

where u_x denotes the embedding of entity/relation x. Typically, $u_x \in \mathbb{R}^d$, but other options are possible. For example, in ComplEx (Trouillon et al. 2016), u_x is complex-valued ($u_x \in \mathbb{C}^d$). ϕ is a scoring function that depends on the link prediction model and will be adressed in more detail in Section 3.1.

Closed-world Link Prediction Closed-world link prediction involves predicting facts about entities on which the link prediction model is trained on. For tail prediction, head and relation are given and the objective is to predict the tail entity with the highest score. This is done by calculating the score of the (h, r) pair with every $t \in E$.

$$tail^* = \arg\max_{t \in E} \; score(h, r, t) \tag{2}$$

Similarly, for head prediction, the score of (r, t) is calculated with every $h \in E$:

$$head^* = \arg\max_{h \in E} \; score(h, r, t) \tag{3}$$

For the remaining part of the paper, we will only discuss the task of tail prediction. The same concepts can be applied to the task of head prediction.

Open-world Extension Open-world link prediction involves predicting facts about entities that the link prediction model was not trained on. Our contribution lies in extending the above graph-based model to perform open-world link prediction. We assume an unseen head entity $head \notin E$ to be represented by its name and textual description, which we concatenated into a word sequence $\mathcal{W} = (w_1, w_2, ..., w_n)$. Word embeddings (such as Word2Vec or

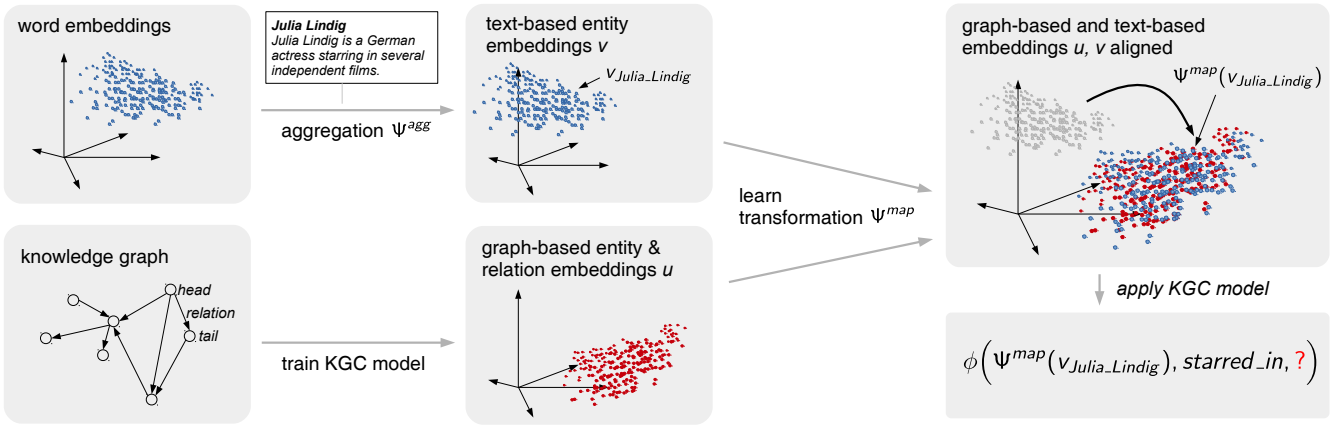


Figure 1: Our approach first trains a KGC model on the graph without using textual information (bottom left). For every entity we can obtain a text-based embedding v by aggregating the word embeddings for tokens in the name and description (top left). A transformation Ψ^{map} is learned on the training entities to map v to the space of graph-based embeddings (right). The learned mapping can then be applied to unknown entities, thus allowing the trained KGC model to be applied.

Glove) pre-trained on a text corpus are then used to transform the sequence of tokens $\mathcal{W}$ into a sequence of embeddings $(v_{w_1}, v_{w_2}, ..., v_{w_n})$. This sequence of embeddings is then aggregated with a function Ψ^{agg} to obtain a text-based embedding of the head entity $v_h \in \mathbb{R}^{d'}$:

$$v_h := \Psi^{agg}\left(v_{w_1}, v_{w_2}, ..., v_{w_n}\right) \tag{4}$$

Since text-based and graph-based embeddings are trained independently on different information sources, we cannot expect them to match. Therefore, a transformation Ψ^{map} is learned from text-based embedding space to graph-based embedding space such that $\Psi^{map}(v_h) \approx u_h$. Then we score triples with the unseen head entity by applying the graph-based model from Equation 1 with the mapped text-based head description:

$$score(h, r, t) = \phi\left(\Psi^{map}(v_h), u_r, u_t\right) \tag{5}$$

The single steps of our model are outlined in more detail in the following.

3.1 Link Prediction Models

Since our approach is independent of the specific link prediction model used, we test three commonly used models in this work:

1. TransE: $\phi(u_h, u_r, u_t) = -||u_h + u_r - u_t||_2$
2. DistMult: $\phi(u_h, u_r, u_t) = \langle u_h, u_r, u_t \rangle$
3. ComplEx: $\phi(u_h, u_r, u_t) = \text{Re}(\langle u_h, u_r, \overline{u}_t \rangle)$

Note that the first two use real-valued embeddings, while ComplEx uses complex-valued embeddings (where $\overline{u} = \text{Re}(u) - i \cdot \text{Im}(u)$ denotes the complex conjugate of embedding u). All models are trained using their original loss functions and validated using closed-world validation data.

3.2 Word Embeddings and Aggregation

We use pre-trained word embeddings trained on large text corpora. Since the number of entities in the datasets used is limited and we found overfitting to be an issue, we omit any refinement of the embeddings. We tested 200-dimensional Glove embeddings (Pennington, Socher, and Manning 2014) and 300-dimensional Wikipedia2Vec embeddings (Yamada et al. 2016).

Note that Wikipedia2Vec embeddings contain phrase embeddings, which we use as an embedding for entity names (like "Julia Lindig"). If no phrase embedding is available, we split the name into single tokens and use token-wise embeddings. If no embedding is available for a token, we use a vector of zeros as an "unknown" token.

To aggregate word embeddings to an entity embedding (function Ψ^{agg}, Equation 4), approaches in the literature range from simple averaging (Pennington, Socher, and Manning 2014) over Long Short Term Memory Networks (LSTMs) (Xu et al. 2017) to relation-specific masking (Shi and Weninger 2017a). We use averaging as an aggregation function. Here, the word embedding vectors are averaged to obtain a single representative embedding. To prevent overfitting, we apply dropout during training, i.e. embeddings of some words are randomly replaced by the unknown token before averaging.

3.3 Transformation Functions

The key to open-world prediction is the mapping from text-based entity embeddings v_e to graph-based ones u_e. Several different transformation functions Ψ^{map} can be learned for this task. In this paper, we discuss three options:

Linear A simple linear function $\Psi^{map}(v) = A \cdot v$. For ComplEx, separate matrices are used for the real and imaginary part: $\Psi^{map}(v) = A \cdot v + i \cdot A' \cdot v$

Affine Here, Ψ^{map} is an affine function $\Psi^{map}(v) = A \cdot v + b$. For ComplEx, separate matrices and vectors are trained just like above: $\Psi^{map}(v) = (A \cdot v + b) + i \cdot (A' \cdot v + b')$

MLP Ψ^{map} is a four layer Multi-Layer Perceptron (MLP) with ReLU activation functions. The output layer is affine. We did not perform an extensive hyperparameter search here.

To train the transformations, first a link prediction model is trained on the full graph, obtaining entity embeddings $u_1, ..., u_n$. We then choose all entities $e_{i_1}, ..., e_{i_m}$ with textual metadata (names and/or descriptions), and extract text-based embedding $v_{i_1}, ..., v_{i_m}$ for them using aggregation (see above). Finally, Ψ^{map} is learned by minimizing the loss function

$$L(\Theta) = \sum_{k=1}^{m} \left|\left| \Psi^{map}_{\Theta}(v_{i_k}) - u_{i_k} \right|\right|_2 \quad (6)$$

using batched stochastic gradient descent, where Θ denotes the parameters of Ψ^{map} (e.g., the weight matrices and bias vectors A, b). For ComplEx, the above loss is summed for real and imaginary parts, and training happens on the sum. We apply no fine-tuning, neither on the graph nor on the text embeddings.

4 Experiments

In this section, we study the impact of our model's parameters (Ψ^{agg}, Ψ^{map}, text embeddings) on prediction performance. We also provide mappings of selected open-world entities, and compare our results with the state-of-the-art.

			Number of Triples		
Dataset	$\|E\|$	$\|R\|$	Train	Valid	Test
FB15k / FB20k	14,904	1,341	472,860	48,991	57,803
DBPedia50k	24,624	351	32,388	123	2,095
FB15k-237-OWE	12,324	235	242,489	12,806	-

Table 1: Dataset statistics for closed-world link prediction.

		Head Pred.		Tail Pred.	
Dataset	$\|E^{open}\|$	Valid	Test	Valid	Test
FB20k	5,019	-	18,753	-	11,586
DBPedia50k	3,636	55	2,139	164	4,320
FB15k-237-OWE	2,081	1,539	13,857	9,424	22,393

Table 2: Dataset statistics for open-world link prediction. E^{open} is the set of novel entities not in the graph. Note that the corresponding closed-world training set is used for training the open-world models.

4.1 Datasets

Closed-world KGC tasks are commonly evaluated on WordNet and Freebase subsets, such as WN18, WN18RR, FB15k, and FB15k-237. For open-world KGC, the following datasets have been suggested: (Xie et al. 2016) introduced FB20k, which builds upon the FB15k dataset by adding test triples with unseen entities, which are selected to have long textual descriptions. (Shi and Weninger 2017a) introduced DBPedia50k and DBPedia500k datasets for both open-world and closed-world KGC tasks.

However, the above datasets display a bias towards long textual descriptions: DBpedia50k has an average description length of 454 words, FB20k of 147 words. Also, for neither of the datasets precautions have been taken to avoid redundant inverse relations, which allows models to exploit trivial patterns in the data (Toutanova and Chen 2015). To overcome these problems, we introduce a new dataset named FB15k-237-OWE. FB15k-237-OWE is based on the well-known FB15K-237 dataset, where redundant inverse relations have been removed. Also, we avoid a bias towards entities with longer textual descriptions: Test entities are uniformly sampled from FB15K-237, and only short Wikidata descriptions (5 words on average) are used.

In the following section, the sampling strategy for FB15k-237-OWE is briefly outlined: For tail prediction test set, we start with FB15K-237 and randomly pick heads (by uniform sampling over all head entities). Each picked head x is removed from the training graph by moving all triples of the form $(x, ?, t)$ to the test set and dropping all triples of the form $(?, ?, x)$ if t still remains in the training set after these operations. Similarly, a head prediction test set is prepared from the set of dropped triplets which satisfy the conditions to be in head prediction test set i.e. head must be represented in training set while tail must not be represented. The dataset also contains two validation sets: A closed-world one (with random triples picked from the training set) and an open-world one (with random triples picked from the test set).

We evaluate our approach on DBPedia50k, FB20k, and the new dataset FB15k-237-OWE. Statistics of the datasets are highlighted in Table 1 and Table 2.

4.2 Experimental Setup

We perform multiple runs using different KGC models, transformation types, training data, and embeddings used. For each run, both KGC model and transformation Ψ^{map} are trained on the training set: the KGC model without using any textual information and the transformation using entity names and descriptions. We manually optimize all hyperparameters on the validation set. Due to the lack of an open-world validation set on FB20k, we randomly sampled 10% of the test triples as a validation set.

Performance Measures Performance figures are computed using tail prediction on the test sets: For each test triple (h, r, t) with open-world head $h \notin E$, we rank all known entities $t' \in E$ by their score $\phi(h, r, t')$. We then evaluate the ranks of the target entities t with the commonly used mean rank (MR), mean reciprocal rank (MRR), as well as Hits@1, Hits@3, and Hits@10.

Note that multiple triples with the same head and relation but different tails may occur in the dataset: $(h, r, t_1), ..., (h, r, t_p)$. Following (Bordes et al. 2013),

Model	DBPedia50k				FB15k-237-OWE				FB20k			
	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR	H@1	H@3	H@10	MRR
Target Filt. Base.	4.5	9.7	23.0	11.0*	6.4	14.2	23.3	12.7	17.5	32.1	41.2	27.2
DKRL	-	-	40.0	23.0	-	-	-	-	-	-	-	-
ConMask	47.1	64.5	**81.0**	58.4*	21.5	39.9	45.8	29.9	42.3	**57.3**	**71.7**	**53.3**
Cmplx-OWE-200	49.0	62.3	73.6	57.7	29.1	41.0	52.7	37.3	44.2	55.9	68.2	52.3
Cmplx-OWE-300	**51.9**	**65.2**	76.0	**60.3**	**31.6**	**43.9**	**56.0**	**40.1**	**44.8**	57.1	69.1	53.1

Table 3: Comparison with other open-world KGC models on tail prediction. Note that we used the same evaluation protocol with target filtering as in ConMask. The asterisk (*) denotes that the result differs from the one published, because the MRR is calculated differently.

when evaluating triple (h, r, t_i) we remove all entities $t_1, ..., t_{i-1}, t_{i+1}, ..., t_p$ from the result list . All results (except *MRR(raw)*) are reported with this *filtered* approach. Note also that when computing the MRR, given a triple (h, r, t_i) only the reciprocal rank of t_i itself is evaluated (and *not* the best out of $t_1, ..., t_i, ..., t_p$, which would give better results). This is common when evaluating KGC models (Bordes et al. 2013) but differs from ConMask's evaluation code, which is why one result in Table 3 differs from (Shi and Weninger 2017a) (see the (*) mark).

Note also that (Shi and Weninger 2017a) add a second filtering method called *target filtering*: When evaluating a test triple (h, r, t), tails t' are only included in the ranked result list if a triple of the form $(?, r, t')$ exists in the training data, otherwise it is skipped. We found this to improve quantitative results substantially, but it limits the predictive power of the model because tails can never be linked via *new* relations. Therefore, we use target filtering only when comparing with the ConMask and DKRL models from (Shi and Weninger 2017a) (Table 3).

Implementation Details For training TransE and DistMult, we use the OpenKE framework[2] which provides implementations of many common link prediction models. For closed-world graph embedding, we use both OpenKE and our own implementation after validating the equivalence of both.

For training the transformation Ψ^{map}, we used the Adam optimizer with a learning rate of 10^{-3} and batch size of 128. For DBPedia50k we use a dropout of 0.5, while for FB20k and FB15k-237-OWE we use no dropout. The embedding used is the pretrained 300 dimensional Wikipedia2Vec embedding and the transformation used is affine unless stated otherwise.

4.3 Comparison with State of the Art

We first compare our model ComplEx-OWE with other open-world link prediction models in Table 3. For a fair comparison, all the results are evaluated using target filtering. For all models and all datasets, 200-dimensional Glove embeddings were used, except for the Complex-OWE300, which uses 300-dimensional Wikipedia2Vec embeddings. The effect of different embeddings will be studied further in Section 4.5.

The results for Target Filtering Baseline, DKRL and ConMask were obtained by the implementation provided by (Shi and Weninger 2017a). The Target Filtering Baseline is evaluated by assigning random scores to all targets that pass the target filtering criterion. DKRL uses a two-layer CNN over the entity descriptions. ConMask uses a CNN over the entity names and descriptions along with the relation-based attention weights.

It can be seen from Table 3 that our best model, ComplEx-OWE300, performs competitively when compared to ConMask. On DBPedia50k, our model performs best on all metrics except Hits@10. On FB20k it is outperformed by a small margin by ConMask but performs better on Hits@1. On FB15k-237-OWE our model outperforms all other models significantly. We believe that this is due to FB15k-237-OWE having very short descriptions. ConMask generally relies on extracting information from the description of entities with its attention mechanism, whereas our model relies more on extracting information from the textual corpus that the word embedding were trained on. This enables our model to provide good results without relying on having long descriptions.

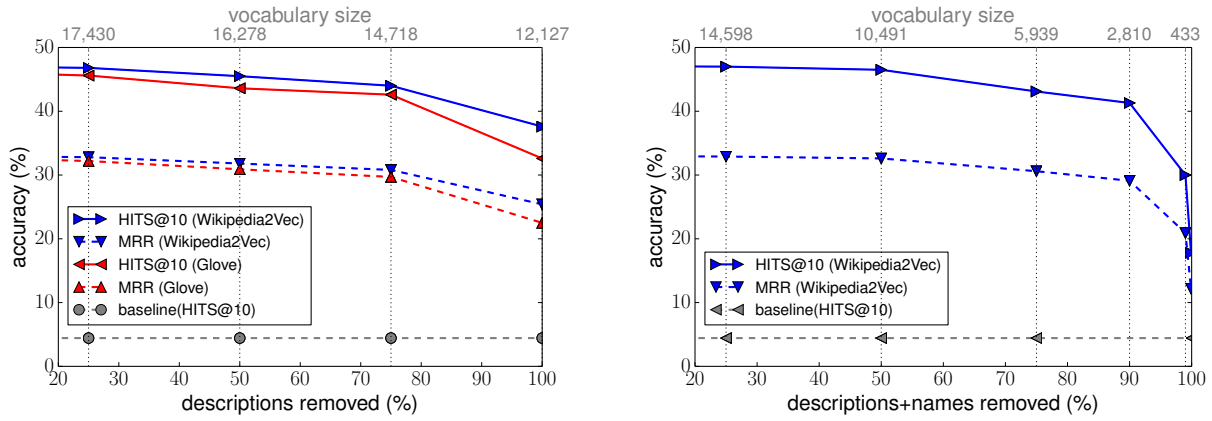
4.4 Analysis of Different Link Prediction Models and Transformations

Our OWE extension for open-world link prediction can be used with any common KGC model. Therefore, we evaluate three commonly used options, namely TransE, DistMult, and ComplEx. Results are displayed in Table 4: All

Model	MRR		HITS@		
	Filt.	Raw	1	3	10
TransE-OWE	28.7	22.9	21.9	31.7	41.0
DistMult-OWE	34.4	25.7	26.6	37.7	**49.2**
ComplEx-OWE	**35.2**	**26.1**	**27.8**	**38.6**	49.1

Table 4: Open-world tail prediction by applying the transformation to different closed-world link prediction models on the FB15k-237-OWE dataset without target filtering.

[2]OpenKE framework: https://github.com/thunlp/OpenKE



(a) Scarce Entity Descriptions

(b) Scarce Entity Metadata (Names and Descriptions)

Figure 2: Performance on FB15k-237-OWE with ComplEx-OWE-300 without target filtering when dropping (a) the entity descriptions or (b) both descriptions and names. The x-axis shows the amount of textual data removed. Even for scarce textual data, learning the transformation Ψ^{map} is robust.

	MRR		HITS@		
Transformation	Filt.	Raw	1	3	10
Linear	33.2	25.2	26.1	36.4	46.5
Affine	**35.2**	**26.1**	**27.8**	**38.6**	**49.1**
MLP	32.2	24.7	25.0	35.3	45.6

Table 5: Comparison of different transformation functions with ComplEx-OWE-300 on the FB15k-237-OWE dataset without target filtering.

three models are trained with embedding dimensionality $d = 300$ on the closed-world dataset. For text embeddings, Wikipedia2Vec embeddings of the same dimensionality were used. It can be seen that the performance on the open-world setting matches the expressiveness of the models: ComplEx-OWE with its ability to model even asymmetric relations yields the best results, while the symmetric DistMult-OWE achieves a similar performance.

We also test different transformation functions Ψ^{map} as illustrated in Table 5. It can be observed that quite simple transformations achieve the strong results: The best performance is achieved by the affine transformation with 49.1% HITS@10 by a margin of 2–4 percent.

4.5 Text Embeddings and Robustness To Missing Entity Metadata

In some cases, the knowledge graph may lack textual metadata (both the name and description) for some or all of its entities. Other models like ConMask and DKRL are dependant on textual descriptions, e.g. ConMask uses attention mechanisms to select relation-specific target words from long texts. Therefore, ConMask and DKRL would require completely dropping triples without metadata and be unable to learn about the link structure of such entities as they use joint training. However, in our approach, we have to drop such entities only during the phase where the transformation Ψ^{map} is learned, while the link prediction model can still be learned on the full graph.

To demonstrate the robustness of our approach to missing entity meta-data, we re-evaluate accuracy when randomly dropping metadata for training entities. Fig. 2 outlines the performance for two scenarios:

- **Dropping descriptions**: We remove only the textual descriptions for a varying percentage of randomly selected entities (between 20% to 100%). The names of these entities are not removed and therefore, we still train Ψ^{map} on them.
- **Dropping all meta-data**: We randomly select entities and remove both their descriptions and names, effectively removing these entities from the training set altogether when training Ψ^{map}.

We also included a baseline experiment to simulate an unsuccessful learning of Ψ^{map}. In this baseline, when evaluating a test triple, we replace its head by the embedding of another random head from the training data. Note that this baseline still gives some reasonable hits for triples where the *relation* is a strong indicator. For example, if we have a triplet $(X, time\_zone, ?)$: Even if the head X is unknown, a model can achieve reasonable accuracy by simply "guessing" time zones as tails.

Overall, Fig. 2 suggests that transformation learning is able to generalize well even with very limited training data. In Fig. 2a only the descriptions of entities have been removed. For Wikipedia2Vec embeddings, this removal has virtually no effect on prediction accuracy. We believe that this is because Wikipedia2Vec embeddings are trained such that we can lookup strong entity embeddings by the name alone. Even when removing 100% of descriptions (i.e., only training on the entity names), accuracy is only 2-3% lower than training on the full graph. However, in case of Glove embeddings, the drop in performance is very significant, especially when the description is dropped for all the entities.

Test Triple	Head Description	Top 4 Nearest Neighbors	Top 4 Predictions
(**Bram Stoker**, /people/person/profession, **Writer**)	Irish novelist and short story writer, best known today for his 1897 Gothic novel Dracula	1. Ursula K. Le Guin 2. Charles Stross 3. Larry Niven 4. Kurt Vonnegut	**1. Writer** 2. England 3. United Kingdom 4. Author
(**Parma**, /.../country, **Italy**)	Italian comune	1. Essen 2. Mantua 3. Bergamo 4. Siena	1. Portugal 2. Netherlands 3. Germany **4. Italy**
(**Bachelor of Science**, /.../institution, **Kingston University)**	Academic degree	1. Masters Degree 2. Doctoral Degree 3. Bachelor of Arts 4. Master of Arts	1. Harvard Law School 2. Wesleyan University 3. Panjab University 4. Baylor University **32. Kingston University**
(**Amtrak**, /.../headquarters.../citytown, **Washington DC)**	Intercity rail operator in the United States	1. AT&T 2. Southwest Airlines 3. Starbucks 4. Delta Airlines	1. Dublin 2. New York City 3. United States Dollar 4. Charlotte **72. Washington DC**

Table 6: Selected results on FB15k-237-OWE. For each test triple (Column 1), the head's name and description (Column 2) is mapped to graph-based embedding space. The nearest training entities in that space (Column 3) indicate a good semantic match. The model predicts reasonable tails, in the first two cases successfully, in the others not.

In Fig. 2b, we remove not only descriptions but also entity names. Even in this case, learning is robust. If half of the $12,324$ training entities are removed, the drop in accuracy is less than 1%. Only when removing 90% of training data (leaving 123 training entities), performance starts to deteriorate significantly. This highlights the ability of our model to learn from a limited amount of training data, when it is important to be able to train the KGC model itself on *all* the entities.

4.6 Selected Results

Finally, we inspect sample prediction results for ComplEx-OWE-300 in Table 6. Besides the final prediction, we also test whether our transformation from text-based to semantic space is successful: For each test triple, we represent the open-world head entity by its text-based embedding v_{head}, match it to a graph-based embedding $\Psi^{map}(v_{head})$, and estimate the nearest neighbor entities in this space. We use the Euclidean distance on the real part of the ComplEx embeddings, but found results to be similar for the imaginary part.

If the transformation works well, we expect these nearest neighbors to be semantically similar to the head entity. This is obviously the case: For *Bram Stoker* (the author of Dracula), the nearest neighbors are other authors of fantasy literature. For *Parma*, the neighbors are cities (predominantly in Italy). For *Bachelor of Science*, the model predicts appropriate entities (namely, Universities) but – even though we apply filtering – the predictions are not rewarded. This is because the corresponding triples, like *(Bachelor of Science, /.../institution, Harward Law School)*, are missing in the knowledge graph.

5 Conclusion

In this work, we have presented a simple yet effective extension to embedding-based knowledge graph completion models (such as ComplEx, DistMult and TransE) to perform open-world prediction. Our approach – which we named OWE – maps text-based entity descriptions (learned from word embeddings) to the pre-trained graph embedding space. In experiments on several datasets (including the new FB15K-237-OWE dataset we introduced in this work), we showed that the learned transformations yield semantically meaningful results, that the approach performs competitive with respect to the state of the art, and that it is robust to scarce text descriptions.

An interesting direction of future work will be to combine our model with approaches like ConMask (Shi and Weninger 2017a), which (1) exploit more complex aggregation functions and (2) use relation-specific attention/content masking to draw more precise embeddings from longer descriptions.

Acknowledgements

This work was partially funded by the German Federal Ministry of Education and Research (Program FHprofUnt, Project DeepCA / 13FH011PX6) and the German Academic Exchange Service (Project FIBEVID / 57402798).

References

Bojanowski, P.; Grave, E.; Joulin, A.; and Mikolov, T. 2017. Enriching Word Vectors with Subword Information. *TACL* 5:135–146.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Adv. in Neural Information Processing Systems*, 2787–2795.

Dong, X. L.; Gabrilovich, E.; Heitz, G.; Horn, W.; Lao, N.; Murphy, K.; Strohmann, T.; Sun, S.; and Zhang, W. 2014. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *Proc. KDD*, 601–610.

Faruqui, M.; Dodge, J.; Jauhar, S. K.; Dyer, C.; Hovy, E. H.; and Smith, N. A. 2015. Retrofitting Word Vectors to Semantic Lexicons. In *NAACL HLT*, 1606–1615.

Ferrucci, D.; Brown, E.; Chu-Carroll, J.; Fan, J.; Gondek, D.; Kalyanpur, A. A.; Lally, A.; Murdock, J. W.; Nyberg, E.; Prager, J.; Schlaefer, N.; and Welty, C. 2010. Building Watson: An Overview of the DeepQA Project. *AI Magazine* 31(3):59–79.

Goldberg, Y. 2016. A Primer on Neural Network Models for Natural Language Processing. *J. Artif. Int. Res.* 57(1):345–420.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. *CoRR* abs/1310.4546.

Nickel, M.; Murphy, K.; Tresp, V.; and Gabrilovich, E. 2016. A Review of Relational Machine Learning for Knowledge Graphs. *Proceedings of the IEEE* 104(1):11–33.

Paulheim, H. 2017. Knowledge Graph Refinement: A Survey of Approaches and Evaluation Methods. *Semantic Web* 8(3):489–508.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global Vectors for Word Representation. In *Proc. EMNLP*.

Peters, M. E.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; and Zettlemoyer, L. 2018. Deep contextualized word representations. In *Proc. NAACL*.

Ristoski, P., and Paulheim, H. 2016. RDF2Vec: RDF Graph Embeddings for Data Mining. In *Proc. ISWC*, 498–514.

Shi, B., and Weninger, T. 2017a. Open-World Knowledge Graph Completion. *CoRR* abs/1711.03438.

Shi, B., and Weninger, T. 2017b. ProjE: Embedding Projection for Knowledge Graph Completion. In *Proc. AAAI*, 1236–1242.

Singhal, A. 2012. Introducing the Knowledge Graph: Things, not Strings. https://googleblog.blogspot.com /2012/05/introducing-knowledge-graph-things-not.html, last retrieved: Aug 2018).

Socher, R.; Chen, D.; Manning, C. D.; and Ng, A. Y. 2013. Reasoning with neural tensor networks for knowledge base completion. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'13, 926–934. USA: Curran Associates Inc.

Toutanova, K., and Chen, D. 2015. Observed Versus Latent Features for Knowledge Base and Text Inference. In *3rd Workshop on Continuous Vector Space Models and Their Compositionality*.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex Embeddings for Simple Link Prediction. In *Int. Conference on Machine Learning*, 2071–2080.

Wang, Z., and Li, J. 2016. Text-enhanced Representation Learning for Knowledge Graph. In *Proc. International Joint Conference on Artificial Intelligence*, 1293–1299.

Xie, R.; Liu, Z.; Jia, J.; Luan, H.; and Sun, M. 2016. Representation Learning of Knowledge Graphs with Entity Descriptions. In *Proc. AAAI*, 2659–2665.

Xu, C.; Bai, Y.; Bian, J.; Gao, B.; Wang, G.; Liu, X.; and Liu, T.-Y. 2014. RC-NET: A General Framework for Incorporating Knowledge into Word Representations. In *Proc. Int. Conf. on Information and Knowledge Management*, 1219–1228.

Xu, J.; Qiu, X.; Chen, K.; and Huang, X. 2017. Knowledge Graph Representation with Jointly Structural and Textual Encoding. In *Proc. Int. Joint Conference on Artificial Intelligence*, 1318–1324.

Yamada, I.; Shindo, H.; Takeda, H.; and Takefuji, Y. 2016. Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation. In *Proc. SIGNLL Conference on Computational Natural Language Learning*, 250–259.

Yang, B.; Yih, W.; He, X.; Gao, J.; and Deng, L. 2014. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. *CoRR* abs/1412.6575.

Yu, M., and Dredze, M. 2014. Improving Lexical Embeddings with Semantic Knowledge. In *Proc. ACL*, 545–550.