PATH RANKING MODEL FOR ENTITY PREDICTION

Xiao Long[*†], *MingHong Yao*[*†], *Liansheng Zhuang*[*], *Houqiang Li*[*], *Shafei Wang*[‡]

[*]University of Science and Technology of China
[†]Peng Cheng Laboratory
[‡]Northern Institute of Electronic Equipment, Beijing, China
longxiao1997@mail.ustc.edu.cn; lszhuang@ustc.edu.cn

ABSTRACT

Knowledge graphs (KGs) often encounter knowledge incompleteness, necessitating a demand for KG completion. Path-based methods are one of the most important approaches to this task. However, since the number of entities is much larger than that of relations in a knowledge graph, existing path-based methods are only used to predict the relations between entity pairs, and are rarely applied to solve the entity prediction task. To address the issue, this paper proposes a new framework called Path Ranking Model (PRM) for the knowledge graph completion task. Our key idea is to exploit both the observable patterns and latent semantic information in relation paths to predict the entities. Extensive experiments on public popular datasets demonstrate the effectiveness of our proposed framework in the entity prediction task.

Index Terms— Knowledge graph, KGC, path ranking

1. INTRODUCTION

Knowledge graphs (KGs) have emerged as an effective way to integrate disparate data sources and model underlying relationships for applications such as natural language processing [1], question answering [2] and recommender systems [3]. They are usually stored in triples of the form (head entity, relation, tail entities), called facts. Typical KGs such as Freebase [4], DBpedia [5] and NELL [6] may contain millions of facts, but they are still far from complete due to the complexity of the real world. For example, 75% of 3 million person entities miss a nationality in Freebase, and 60% of person entities do not have a birthplace in DBpedia. Such knowledge graphs are difficult to use in real applications because of no correct answers for questions based on incomplete knowledge graphs. Knowledge graph completion (KGC) [7] tries to solve this incompleteness by inferring new triples based on the existing ones, and has attracted much attention recently.

This work was supported in part by Next Generation AI Project of China No.2018AAA0100602, in part to Dr. Liansheng Zhuang by NSFC under contract No.U20B2070 and No.61976199, and in part to Dr. Houqiang Li by NSFC under contract No.61836011. Dr. Liansheng Zhuang is the corresponding author.

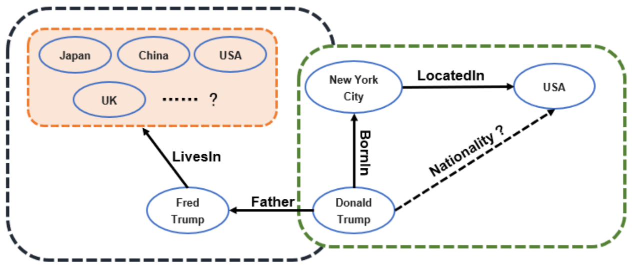

Fig. 1. An example of two forms of KGC tasks. The left side of the figure above describes the entity prediction task and the right side describes the relation prediction task.

Many methods have been proposed to infer the incompleteness of a knowledge graph, among which knowledge graph embeddings (KGEs) are popular ones. The KGEs such as TransE [8], DistMult [9] and RotatE [10] often learn low-dimensional representations of the entities and relations in a knowledge graph, and provide a generalizable context about the overall KG that can be used to infer relations. The learned embeddings contain rich semantic information and can benefit a broad range of downstream applications [1, 2]. However, the KGEs models purely treat the triples in a KG independently and heavily rely on their ability to model connectivity patterns of the relations.

Many efforts have been devoted to path-based methods which aim to use paths to infer relations between an entity pair [11, 12]. Different from KGEs which model the direct path between an entity pair, multi-hop paths are the paths via multi nodes between the source and destination nodes. When routing between two nodes, a multi-hop path that consists of a sequence of triples can provide more informative knowledge than a direct path. For example, let us consider two triples (Donald Trump, BornIn, New York City) and (New York City, LocatedIn, the USA), which constitute a path Donald Trump $\longrightarrow$ BornIn $\longrightarrow$ New York City $\longrightarrow$ LocatedIn $\longrightarrow$ USA via the intermediate node New York City. Then, the machines are supposed to infer over the multi-hop path to conclude that missing relation Nationality exists between

Donald Trump and the USA. In order to make full use of the relation path information in KG, many works are devoted to mining observable patterns in knowledge graphs for reasoning. In early path-based methods such as PRA [11], each path is treated as an atomic feature and used to predict the relation in a binary classifier. As a result, there used to be millions of distinct paths in a single classifier, not to mention that the size increases dramatically with the number of relations in a KG. To solve this problem, Path-RNN [12] takes multi-hop relation paths as input for RNN to construct a vector representation for the path. After that, the predictability of the path to a query relation is calculated by dot-product on their representations.

Though achieving promising performance, path-based methods are only used to predict the relationship between an entity pair. Since the number of entities in KG is far greater than the number of relations, it is more challenging to use path-based methods to predict entities that to predict relations. The Fig. 1 illustrates the differences between relation prediction and entity prediction. In fact, multi-hop paths contain rich semantic cues and are extremely useful for entity prediction tasks. For example, through the knowledge contained in this path (Donald Trump $\longrightarrow$ BornIn $\longrightarrow$ New York City $\longrightarrow$ LocatedIn $\longrightarrow$ USA), one can be confident that Donald Trump's father also lives in the USA. Because the relation path (BornIn $\longrightarrow$ LocatedIn) determines the tail entity selection under the path (Father $\longrightarrow$ LivesIn).

Inspired by the above insights, this paper proposes a novel framework to solve the entity prediction task, namely Path Ranking Model (PRM). Our key idea is to use the latent semantic information and observable patterns in relation paths for entity prediction tasks in the knowledge graph. Specifically, our framework first uses the depth-first search (DFS) to find all the effective paths (meta-paths) to represent for each relation. Then, we represent the paths into continuous vector space. Finally, by combining the path features linearly and activating them, we will get the probability that this triple appears in the KG. Extensive experiments on three popular benchmark datasets demonstrate the effectiveness of the PRM in entity prediction tasks.

In summary, our main contributions are listed as follows:

- This paper proposes a novel PRM framework for entity prediction task, which simultaneously exploits both the latent semantic information and observable patterns in relation paths. To the best of our knowledge, this is the first work to use the path-based methods to solve the entity prediction task.

- Extensive experiments are conducted on three popular benchmark datasets. Experimental results show that the PRM achieves the state-of-the-art performance in the entity prediction task.

2. RELATED WORK

In this section, we will describe the related work and the key differences between them and our works. Roughly speaking, we can divide knowledge graph completion models into two categories—KG embedding models and path-based models.

KG embedding models: Knowledge graph embedding, which aims to represent entities and relations as low dimensional vectors. It often fall into three major categories: (1) Translational distance model: Inspired by the translation invariant principle from word embedding [13]. TransE [8] describes relations as translations from head entities to tail entities, which means that entities and relations satisfy the formula $\boldsymbol{h} + \boldsymbol{r} \approx \boldsymbol{t}$, where $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t} \in \mathbb{R}^n$. (2) Tensor decomposition based model: DistMult [9] and ComplEx [14] both use tensor decomposition to represent each relation as a matrix and each entity as a vector. (3) Neural network based models: ConvE [15] employs convolutional neural networks to define score functions. Recently, graph convolutional networks are also introduced, as knowledge graphs have graph structures [16]. However, most of KGE methods merely consider the facts immediately observed in KG and ignore extra prior knowledge to enhance KG embedding.

Path-based models: Paths existing in KG have gained more attention to be combined with KG embedding because multi-hop paths could provide relations between seemingly unconnected entities in KG. Path Ranking Algorithm (PRA) [11] is one of the early studies which searches paths by random walk in KG and regards the paths as features for a per-target relation binary classifier. [12] proposed a compositional vector space model with a recurrent neural network to model relational paths on knowledge graph completion. [17] proposed PTransE to obtain the path embeddings by composing all the relations in each path.

Compared with KG embedding models, path-based models fully consider the diversity of relation paths in KG and capture the rich information in them. However, the previous studies only verified the effectiveness of the path-based methods can decide whether the given query relation exists between the entity pairs, Ignoring the entity prediction task.

3. METHOD

3.1. Overview of the PRM

In this subsection, we will generally introduce the PRM framework and the details of the feature generation. When the traditional path ranking algorithm predicts an unknown relationship, it usually firstly extracts the useful meta-paths, which can be used to represent a relation. And then it generates the paths' features through random walk. However, this way will be very slow for large knowledge graph. Therefore, the proposed PRM can be divided into two stages, online and offline. The offline stage will store the corresponding paths'

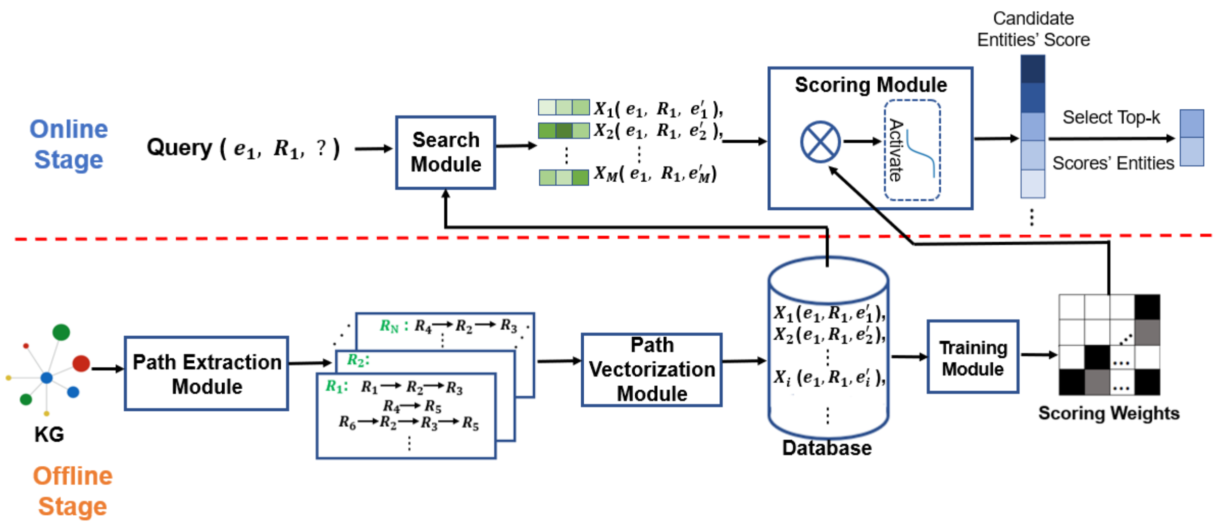

Fig. 2. Pipeline of the PRM on solving the entity prediction task.

features and scoring weights, while the online stage will use them for entity prediction task. Fig. 2 illustrates the pipeline of the PRM on how to solve the entity prediction task.

Path Extraction Module

Given a knowledge graph, we can formulate one fact triple as (e, R, e'). $T = \{(e_i, R_i, e_i'), \cdots\}$ is the set of all triples in KG. And we can also write (e, R, e') as $R(e, e')$ if e and e' are related by R. A relation path P is a sequence of relations $R_1, \cdots, R_\ell$ with constraint that $\forall i : 1 < i < \ell - 1$. For each relation R_i in KG, we use the depth-first search (DFS) to find the meta-paths $\mathcal{P} = \{P_1, \cdots, P_n\}$ of the bounded length ℓ through all triples, which means the maximum length of the meta-paths is ℓ. In fact, ℓ is a hyper-parameter that needs to be adjusted.

Path Vectorization Module

In order to emphasize the types associated with each step, path $P = R_1, \cdots, R_\ell$ can be written as:

$$E_0 \xrightarrow{R_1} \cdots \xrightarrow{R_\ell} E_\ell, \tag{1}$$

where $E_i = range(R_i) = dom(R_{i+1})$, using $dom(R_i)$ to denote the domain of R, and $range(R)$ for its range. And we also define $dom(P) \equiv E_0$, $range(P) \equiv E_\ell$. One can recursively define a distribution, denoted as $h_{E_e,P}(e')$, to describe the probability that the head entity e is connected to the tail entity e' through the path P. For any relation path $P = R_1, \cdots, R_\ell$, and set of seed entities $E_e \subset dom(P)$, one can define the following distribution if P is the empty path:

$$h_{E_e,P}(e') = \begin{cases} 1/|E_e|, & \text{if } e' \in E_e, \\ 0, & \text{otherwise .} \end{cases} \tag{2}$$

If $P = R_1, \cdots, R_\ell$ is nonempty, then let $P' = R_1, \cdots, R_{\ell-1}$, and define:

$$h_{E_e,P}(e') = \sum_{q \in range(P')} h_{E_q,P'}(q) \cdot P(e \mid q; R_\ell), \tag{3}$$

where $P(e|q; R_\ell) = \frac{R_\ell(q,e)}{|R_\ell(q,\cdot)|}$, is the probability of reaching node e from node q a one step random walk with edge type R_ℓ, $R(e', e)$ indicates whether there exists an edge with type R that connect q to e.

More generally, given a set of paths $P_1, \cdots, P_n$, one could treat each $h_{E_e,P_i}(e')$ as a path feature for the node e, and rank nodes by a linear model

$$\theta_1 h_{E_e,P_1}(e') + \theta_2 h_{E_e,P_2}(e') + \cdots + \theta_n h_{E_e,P_n}(e'), \tag{4}$$

where θ_i are weights for the paths. In this paper, we consider learning such linear weighting schemes over all relation paths of bounded length ℓ (e.g., $\ell \leq 4$). One can easily generate $\mathcal{P}(e, l) = \{P\}$, the set of all type-correct relation paths with range T_e and length $\leq l$. This gives a ranking of nodes $e' \in I(T_e)$ by the following scoring function

$$s(e'; \theta) = \sum_{P \in \mathcal{P}(e,l)} h_{E_e,P}(e')\theta_P. \tag{5}$$

In matrix form this could be written $s = A\theta$, where s is a sparse column vector of scores, and θ is a column vector of weights for the corresponding paths P. We will call A the feature matrix, and denote the i-th row of A as A_i.

Finally, we will store the corresponding paths' features in the database, which can accelerate the speed of the prediction in online stage.

Training Module

Given a relation R and a set of node pairs $\{(e_i, e_i')\}$, we can construct a training dataset $D = \{(\mathbf{x}_i, l_i)$ by (3) and

(4), where $\mathbf{x}_i$ is a vector of all the path features for the pair $\{(e_i, e'_i)\}$ i.e., the j-th component of $\mathbf{x}_i$ is $h_{e_i,P_j}(e'_i)$, and l_i indicates whether $R(e_i, e'_i))$ is true. It can be formulated as follows:

$$l_i = \begin{cases} 1, & \text{if } (e_i, R, e'_i) \in T \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

With the training data D, we formalize $o_i(\theta)$ as the per-instance objective function like (7). Parameter θ is the weights to be estimated. In this paper, we use binomial log-likelihood, which has the advantage of being easy to optimize and also does not penalize outlier samples too harshly [18].

$$o_i(\theta) = w_i \left[l_i \ln p_i + (1 - l_i) \ln (1 - p_i)\right] \tag{7}$$

where p_i is the probability of triple appearing in KG, it can be calculated by (8):

$$p\left(l_i = 1 \mid \mathbf{x}_i; \theta\right) = \frac{\exp\left(\theta^T \mathbf{x}_i\right)}{1 + \exp\left(\theta^T \mathbf{x}_i\right)} \tag{8}$$

So, the parameter estimation can be formulated as maximizing a regularized objective function:

$$O(\theta) = \sum_i o_{(i)}(\theta) \tag{9}$$

where λ is a parameter controls $L2$-regularization to prevent overfitting. After the training stage, we can get the scoring weights for online prediction.

3.2. PRM For Entity Prediction Task

In this subsection, we will describe how to formulate entity prediction on a knowledge graph as a ranking task. Since lacking head entity or tail entity are equivalent, here we only discuss the latter.

Given a triple that lacks tail entity $(e_i, R_i, ?)$. We use the all candidate entities in KG to expand to a set of triples $\mathcal{T} = \{(e_i, R_i, e'_1), \cdots, (e_i, R_i, e'_M)\}$, where M is the number of the entities in KG. Then, we retrieve the paths' feature $\mathbf{x}_i$ of the corresponding triples in the database generated in the offline stage. Next, we can calculate the probability of all candidate entities p_i through the $\mathbf{x}_i$ by (8).

$$ind = \arg\max_i \{p_i\} \tag{10}$$

Finally, selecting the entity index ind with the highest score by (10), the e'_{ind} is the most likely tail entity to be predicted.

4. EXPERIMENT

In this section, we evaluate the performance of the PRM on the task of entity prediction. All the algorithms are implemented by python and PyTorch, and run the experiments on 2 Quadro RTX 6000 and 96 Intel Xeon Platinum 8268.

4.1. Experimental Settings

Datasets. We evaluate the PRM on three popular benchmark datasets in the task of KGC. FB15k-237 [7], YAGO3-10 [7], and WN18RR [7]. They are subsets of WN18, FB15k, and YAGO3 respectively. Recent studies [7] found that the FB15k and WN18 contain inverse relations. Under this circumstances, FB15k-237 and WN18RR were proposed, which removed the reverse relations in FB15k and WN18. So, they are regarded as more challenging datasets. Therefore, we use FB15k-237, WN18RR, and YAGO3-10 as the benchmark datasets. The statistics of the datasets are summarized in Table 1.

Table 1. Statistics of the Three Datasets. Rel denotes relation and Ent denotes entity.

Dataset	#Ent	#Rel	#Train	#Valid	#Test
FB15k-237	14,541	237	272,115	17,535	20,466
YAGO3-10	123,182	37	1,079,040	5,000	5,000
WN18RR	40,493	11	86,835	3,034	3,134

Baselines. To demonstrate the effectiveness of the PRM in link prediction task, we select several involved state-of-the-art models for comparison, including two types of baselines: (1) KGE methods, which has been widely used in entity prediction tasks, such as: TransE [8], DistMult [9], ComplEx [14], RotatE [10], ConvE [15], R-GCN [16], HAKE [19], InteractE [20] and TuckER [21]. (2) Rule enhanced models, which aims to use first-order logic rules in KG for reasoning [7] like: DRUM [22]. We use the best results presented in their original papers for comparison.

Evaluation Metric. Following Bordes [8], for each triple (h, r, t) in the test dataset, we replace either the head entity h or the tail entity t with each candidate entity to create a set of candidate triples. We then rank the candidate triples in descending order by their scores. It is worth noting that we use the "Filtered" setting as in [8], which does not take any existing valid triples into accounts at ranking. We choose Mean Reciprocal Rank (MRR, the mean of all the reciprocals of predicted ranks) and Hits at N (H@N, the proportion of ranks not larger than N) as the evaluation metrics. Higher MRR or H@N indicates better performance.

Parameters Setting. During the training stage, we use the adaptive moment (Adam) algorithm to optimize the model. And we search the best hyper-parameters of all models according to the performance on the validation set. Notice that the validation set does not participate in training. In detail, we search the learning rate α in {0:001; 0:005; 0:01; 0:1}, meta-paths length ℓ for one relation in {2; 3; 4}. Finally, we set the α: 0.01, ℓ: 4, maximum numbers of meta-paths under each relation: 500 for all the datasets. And all the training parameters are randomly initialized.

Table 2. Entity prediction results on FB15k-237, WN18RR.

	FB15K-237				WN18RR			
	MRR	Hits@N			MRR	Hits@N		
		@1	@3	@10		@1	@3	@10
TransE(Bordes et al. 2013)	.294	-	-	.465	.226	-	-	.501
DistMult(Yang et al. 2015)	.241	.155	.263	.419	.430	.391	.442	.490
ComplEx(Trouillon et al. 2016)	.247	.158	.275	.428	.440	.158	.275	.428
R-GCN(Schlichtkrull et al. 2018)	.249	.151	.264	.417	.249	.151	.264	.417
ConvE(Dettmers et al. 2018)	.316	.239	.350	.491	.461	.390	.430	.481
RotatE(Sun et al. 2019)	.338	.241	.375	.533	.476	.428	.492	.571
DRUM(Sadeghian et al.2019)	.343	.255	.378	.516	.486	.425	.513	.586
TuckER(Balazevic et al.2019)	.358	**.266**	**.394**	.544	.470	.443	.482	.526
HAKE(Zhang et al.2020)	.346	.250	.381	.542	.497	**.452**	.516	.582
InteractE(Vashishth et al.2020)	.354	.263	-	.535	.463	.430	-	.528
PRM	**.364**	.255	.388	**.580**	**.498**	.431	**.733**	**.766**

Table 3. Entity prediction results on YAGO3-10

	YAGO3-10			
	MRR	Hits@N		
		@1	@3	@10
TransE(Bordes et al. 2013)	-	-	-	-
DistMult(Yang et al. 2015)	.340	.240	.380	.540
ComplEx(Trouillon et al. 2016)	.360	.260	.400	.550
ConvE(Dettmers et al. 2018)	.520	.450	.560	.660
RotatE(Sun et al. 2019)	.495	.402	.550	.670
HAKE(Zhang et al. 2020)	.545	.462	.596	.694
InteractE(Vashishth et al. 2020)	.541	.462	-	.687
PRM	**.698**	**.526**	**.692**	**.723**

4.2. RESULTS AND ANALYSIS

Experimental results are shown in Table 2 and Table 3. From Table 2 and Table 3, we have the following findings.

1) The results indicate that PRM significantly and consistently outperforms all the state-of-the-art competitors on three benchmark datasets. In both Table 2 and Table 3, PRM achieves the best results on most metrics. The experimental results clearly demonstrate the effectiveness of the PRM in the entity prediction task.

2) Specifically, in the YAGO3-10 dataset, PRM is 15.3% higher than HAKE's MMR, 6.4% higher than InteractE's Hits@1, 9.6% higher than HAKE's Hits@3, and 2.9% higher than HAKE's Hits@10. In the WN18RR dataset, PRM improves upon DRUM's Hits@10 and Hits@3 by a large margin of 18% and 22% respectively respectively. And the Hits@1 and MRR are basically the same compared to other methods. In the FB15K237 dataset, PRM also has achieved good results on all evaluation metrics. Although on FB15k-237, the performance of the Hit@1 and Hit@3 is not so good, we can still observe that PRM is better than KGE methods on MMR metrics all the time. This also explains that PRM can guarantee the average accuracy of the prediction results. The experimental results illustrate that the path-based model can capture both the latent semantic information and observable patterns in KG, which has great help for the entity prediction task.

Due to the differences between the FB15k-237 dataset and the other two datasets, the performance of different types of algorithms may be different. For example, the PRM performs much better on YAGO3-10 dataset and WN18RR dataset than KGE models. However, on the FB15k-237 dataset, the improvement to the entity prediction task by PRM is not so significant. A possible reason is the number of relations in FB15k-237 is much more than the other two datasets, which means there are more complex relation path patterns and less training data under one relation. Besides, from the above experimental results, it can be found that the path-based method can achieve greater improvement on Hits@N ($N > 1$) than Hits@1, which means the path-based method can usually predict the best solution within the tolerable error range.

5. CONLUSION

In this paper, we first propose the PRM framework to solve the entity prediction task. Compared with KGE models treating the triples independently, and only using the direct relations between entities. PRM simultaneously exploits both the latent semantic information and observable patterns in relation paths, which has great help for predicting the missing entities in KG. The extensive experimental results verified the paths' features with different weights learned by PRM are significant in improving the performance on the entity prediction task. In the future, we will investigate how to use structure information and more entity information (e.g., entity type.) to help the path-based model get better results.

6. REFERENCES

[1] Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu, "Ernie: Enhanced language representation with informative entities," *arXiv preprint arXiv:1905.07129*, 2019.

[2] Yanchao Hao, Hao Liu, Shizhu He, Kang Liu, and Jun Zhao, "Pattern-revising enhanced simple question answering over knowledge bases," in *Proceedings of the 27th International Conference on Computational Linguistics*, 2018, pp. 3272–3282.

[3] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 353–362.

[4] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1247–1250.

[5] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al., "Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia," *Semantic web*, vol. 6, no. 2, pp. 167–195, 2015.

[6] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, Bishan Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi, Matt Gardner, Bryan Kisiel, et al., "Never-ending learning," *Communications of the ACM*, vol. 61, no. 5, pp. 103–115, 2018.

[7] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

[8] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko, "Translating embeddings for modeling multi-relational data," *Advances in neural information processing systems*, vol. 26, pp. 2787–2795, 2013.

[9] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv preprint arXiv:1412.6575*, 2014.

[10] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang, "Rotate: Knowledge graph embedding by relational rotation in complex space," *arXiv preprint arXiv:1902.10197*, 2019.

[11] Ni Lao, Tom Mitchell, and William Cohen, "Random walk inference and learning in a large scale knowledge base," in *Proceedings of the 2011 conference on empirical methods in natural language processing*, 2011, pp. 529–539.

[12] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum, "Compositional vector space models for knowledge base completion," *arXiv preprint arXiv:1504.06662*, 2015.

[13] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[14] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard, "Complex embeddings for simple link prediction," International Conference on Machine Learning (ICML), 2016.

[15] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel, "Convolutional 2d knowledge graph embeddings," *arXiv preprint arXiv:1707.01476*, 2017.

[16] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference*. Springer, 2018, pp. 593–607.

[17] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu, "Modeling relation paths for representation learning of knowledge bases," *arXiv preprint arXiv:1506.00379*, 2015.

[18] Karl Breitung, "Probability approximations by log likelihood maximization," *Journal of engineering mechanics*, vol. 117, no. 3, pp. 457–477, 1991.

[19] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang, "Learning hierarchy-aware knowledge graph embeddings for link prediction," in *Thirty-Fourth AAAI Conference on Artificial Intelligence*. 2020, pp. 3065–3072, AAAI Press.

[20] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Nilesh Agrawal, and Partha Talukdar, "Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions," in *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. 2020, pp. 3009–3016, AAAI Press.

[21] Ivana Balažević, Carl Allen, and Timothy M Hospedales, "Tucker: Tensor factorization for knowledge graph completion," *arXiv preprint arXiv:1901.09590*, 2019.

[22] Ali Sadeghian, Mohammadreza Armandpour, Patrick Ding, and Daisy Zhe Wang, "Drum: End-to-end differentiable rule mining on knowledge graphs," in *Advances in Neural Information Processing Systems*, 2019, pp. 15347–15357.