KGRefiner: Knowledge Graph Refinement for Improving Accuracy of Translational Link Prediction Methods

Anonymous ACL submission

Abstract

The Link Prediction is the task of predicting missing relations between entities of the knowledge graph. Recent work in link prediction has attempted to provide a model for increasing link prediction accuracy by using more layers in neural network architecture. In this paper, we propose a novel method of refining the knowledge graph so that link prediction operation can be performed more accurately using relatively fast translational models. Translational link prediction models, such as TransE, TransH, TransD, have less complexity than deep learning approaches. Our method uses the hierarchy of relationships and entities in the knowledge graph to add the entity information as auxiliary nodes to the graph and connect them to the nodes which contain this information in their hierarchy. Our experiments show that our method can significantly increase the performance of translational link prediction methods in H@10, MR, MRR.

1 Introduction

Knowledge graphs represent a set of interconnected descriptions of entities, including objects, events, or concepts. These graphs are structures by which knowledge is stored in triples. These triples include the three parts head, relation, and tail. The relation determines the type of relationship between head and tail. These graphs are becoming a popular approach to display and model different information in the world. Additionally, knowledge graphs have several applications, for example, question answering systems (Bordes et al., 2014a,b), recommendation systems (Zhang et al., 2016), search engines (Xiong et al., 2017), relationship extraction (Mintz et al., 2009), etc.

Despite many efforts to build knowledge graphs, they are not complete yet. For example, in the Freebase (Bollacker et al., 2008), over 70% of people do not have their place of birth in the graph. This incompleteness of knowledge graphs has motivated researchers to add information to the graph and complete it.

One of the developing fields in completing the knowledge graph is knowledge graph embedding (KGE). The task of KGE is to embed entities and relationships in a small continuous vector space. One application of these embedding is to predict missing links in the knowledge graph.

Translational link prediction models use the sum of the head and relation vectors to predict the tail. These models started with TransE (Bordes et al., 2013), and after that, TransH (Wang et al., 2014), TransR (Lin et al., 2015), TransD (Ji et al., 2015), RotatE (Sun et al., 2019), etc., tried to improve it in the following years. The advantages of translational methods over deep learning techniques are that they are robust, and their score function is considerably faster. Therefore, in this work, we tried to improve these translational methods.

There is a lot of information in knowledge graphs. The hierarchy of entities and relationships is part of it. Paris, for example, its hierarchy is "entity → physical_entity → object → location → region → area → center → seat → capital → national_capital". This hierarchy is not given enough attention in link prediction methods, and we intend to use this information in this paper.

SACN (Shang et al., 2019) added some nodes and relationships to the graph to use the graph structure information but did not justify adding these nodes and edges, so it is not generalizable for other graphs. In addition, SACN added this information only to FB15k-237 and did not provide a method for WN18RR. In this paper, we added a much smaller number of relationships and fewer nodes to the graph training section by interpreting them. HRS (Zhang et al., 2018) used relation clusters and sub-relations to use this information. Nevertheless, like SACN, this can not be generalized well.

The ETE (Moon et al., 2017) considered that if two entities are embedded closely in the embedding space, they are similar and assigned entities' classes based on closeness. Still, our assumption is if two entities use the same relation in the graph or have common elements in their hierarchies, they are related.

When link prediction models learned the relation between Paris and France, previous link prediction methods did not notice that Paris is a city and France is a country. To use this information, we added auxiliary nodes to the graph that included the classes of entities and connected them to related entities. For example, we added an extra node for countries to the knowledge graph and connected it to all the knowledge graph countries. Our contributions are as follows:

- We presented a method for refining the knowledge graph, which is independent of the structure of the link prediction model and adds triples to the knowledge graph. These triples increase the accuracy of translational link prediction with the same time and space complexity of translational models.
- We evaluated our proposed method on two FB15k-237 and WN18RR datasets with successful translational models. The results showed that accuracy in link prediction was significantly increased on H@10, MRR, and MR.

2 Related Work

Knowledge graph embedding is an active and developing field to embed the entities and relations of the knowledge graph. These embeddings are used in link prediction, question answering systems, relation extraction, etc. Knowledge graph link prediction starts with TransE (Bordes et al., 2013), which is the first translational link prediction method. It interprets relation as a transition from head entity to tail in the graph. Some drawbacks of the TransE model are its inability to model N-1, 1-N, and N-N relationships. In the following years, some other translational approaches, such as TransH (Wang et al., 2014), TransD (Ji et al., 2015), and TransR (Lin et al., 2015), were inspired by the initial idea of TransE (Bordes et al., 2013) and tried to improve it. These translational models have much more speed against deep learning models such as ConvE (Dettmers et al., 2018), ConvKB (Nguyen et al., 2018), SACN (Shang et al., 2019), and HAKE (Zhang et al., 2020), but their accuracy is slightly lower than these models. Therefore, we proposed a method to increase the accuracy of these translational models.

Knowledge graph refinement is a field of correcting or improving the knowledge graph. BioKG (Zhao et al., 2020), which worked on medical graphs, has tried to provide a method for removing the wrong information in these graphs. Other works in the refinement of the knowledge graphs try to add information. SACN (Shang et al., 2019) has also added attributes to the knowledge graph, like our work. SACN proposed FB15k-237_Attr; this method for constructing this dataset has three major issues. First, it only worked for FB15k-237, but our proposed method can be applied on WN18RR as well. Second, it has brought the number of FB15k-237 relations from 237 to 484; therefore, it brings more time complexity than ours. However, we only proposed two new relations for FB15k-237 and only one relation for WN18RR. Third, these new relations and entities are not interpretable in SACN; It does not provide a reason for adding these attributes. So it can not be generalized on other graphs.

HRS (Zhang et al., 2018) tried to use sub-relation and relation-cluster to make better predictions. It used the hierarchy of relations as a sub-relationship, and it created a relation cluster to use these as two additional parts of the transition in the translational models. Because links in Wordnet do not have information about entities, HRS sub-relation and relation-cluster on Wordnet are meaningless.

GrCluster (Ranganathan et al., 2020) used path similarity over entities in Wordnet and slightly improved link prediction accuracy. Nonetheless, GrCluster only improved WNNH and WN18, which are not standard link prediction datasets (Dettmers et al., 2018).

3 Background

Suppose E as the collection of all entities of knowledge graph and R set of all its relationships. The (e_s, r, e_o) is called a triple. The $e_s \sim$ E is the head, and $e_o \sim E$ is the tail of a triple. Finally, $r \sim E$ represents the relation between e_s and e_o.

3.1 Link Prediction

Link prediction is the task of predicting the missing link of a knowledge graph by inferring from existing facts on it. The score function of link prediction methods is $\psi(e_o, r, e_s)$, which evaluates triple's accuracy. Our goal in teaching a model that has the highest estimation for the missing triplets of the graph and the lowest prediction for false triples.

3.2 Translational Link Prediction Models

Translational link prediction methods consider the relation as a transition from head to tail. For example (Paris, Capital of, France), the relation "Capital of" is a transition from Paris to France. TransE (Bordes et al., 2013) is the first translational link prediction model. In TransE, embeddings for correct triples are learned as $e_s + r \sim e_o$. It means that the sum of the head's embedding and relation's embedding must be close to the tail; primarily, the distance measure is the L2 norm. Here are some translational link predictions:

TransE: For factual triple (e_s,r,e_o), adding embeddings of head and relation should be closed to the tail embedding, and on the other hand, for corrupted ones (e_s,r,$e_o\prime$), e_s+r should have a distance with $e_o\prime$. The score function of TransE is as follow:

$$\psi(e_o, r, e_s) = -||h + r - t||_2^2$$

TransH (Wang et al., 2014)**:** To improve modelling of N-1, 1-N and N-N, TransH defined a hyperplane for each relations, and translation property should be established on that hyperplane.

$$h_\perp = w_r^\perp h w_r \ , \ t_\perp = w_r^\perp t w_r$$
$$\psi(e_o, r, e_s) = -||h_\perp + r - t_\perp||_2^2$$

TransD (Ji et al., 2015) **:** It creates a dynamic matrix for all entity-relation pairs and maps the head and tail into M1 and M2, respectively. The transition from head to tail is as follow:

$$M_r^1 = w_r w_h^\perp + I \ , \ M_r^2 = w_r w_t^\perp + I$$
$$h_\perp = M_r^1 h \ , \ t_\perp = M_r^2 t$$
$$\psi(e_o, r, e_s) = -||h_\perp + r - t_\perp||_2^2$$

TransR (Lin et al., 2015) **:** It considers that entities may have multiple aspects, and various relations focus on different aspects of entities. It projects entities into relation space by projection matrix M.

$$h_\perp = M_r h \ , \ h_\perp = M_r t$$
$$\psi(e_o, r, e_s) = -||h_\perp + r - t_\perp||_2^2$$

RotatE (Sun et al., 2019) **:** RotatE deals with relation as a rotation to complex space. This rotation brings the source entity to the target entity in the complex space. The relation applies to the head entity by Hadamard product. Then it uses the L1 norm to measure the distance from the tail entity in the score function.

$$\psi(e_o, r, e_s) = -||h_\circ r - t_\perp||^2$$

3.3 Knowledge Graph Refinement

The knowledge graph refinement follows two main objectives: (A) adding information to the knowledge graph, a subcategory of the knowledge graph completion. (B) Detecting incorrect information and removing those triplets from the knowledge graph to increase the correctness of the knowledge graph (Pipino et al., 2002). In this work, we add information to enrich the graph.

4 KGRefiner

In this work, we propose a method to add information to the graph, which refines the knowledge graph and increases link prediction accuracy. In FB15k-237, we do this refinement by using relation hierarchies, and in WN18RR, we use hierarchies of entities. We add this information to the graph as a new node; these nodes are auxiliary nodes. We introduce several new relations to connect these new nodes to graph nodes, and we add these triples to the graph.
Translational link prediction methods such as TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransD (Ji et al., 2015), etc., create transition property in their embeddings. For example, in TransE, embeddings are made as follow:

$$e_s + r \approx e_o \tag{1}$$

This means in embedding space; the tail entity should be close to the sum of head and relation. For example, let's consider these triples:

$$Paris + capitalof \approx France \tag{2}$$
$$Tehran + capitalof \approx Iran \tag{3}$$

Link prediction model is not aware of both tails entities are country. If we add new node as "country" to the graph and connect it to all graph's countries with a new relation "RelatedTo" then these triples are added to graph:

$$France + RelatedTo \approx country \tag{4}$$
$$Iran + RelatedTo \approx country \tag{5}$$

Dataset	FB15k-237	FB15k-237-Refined	WN18RR	WN18RR-Refined	FB15k-237-Attr
Entities	14541	14826	40943	41150	14744
Relations	237	239	11	12	484
Train Edges	272115	550998	86835	230135	350449
Val. Edges	17535	17535	3034	3034	17535
Test Edges	20466	20466	31134	31134	20466

Table 1: Statistics of the experimental datasets. The refined version represents that graph has some auxiliary nodes. These auxiliary nodes are extracted from entities hierarchy in the original knowledge graph.

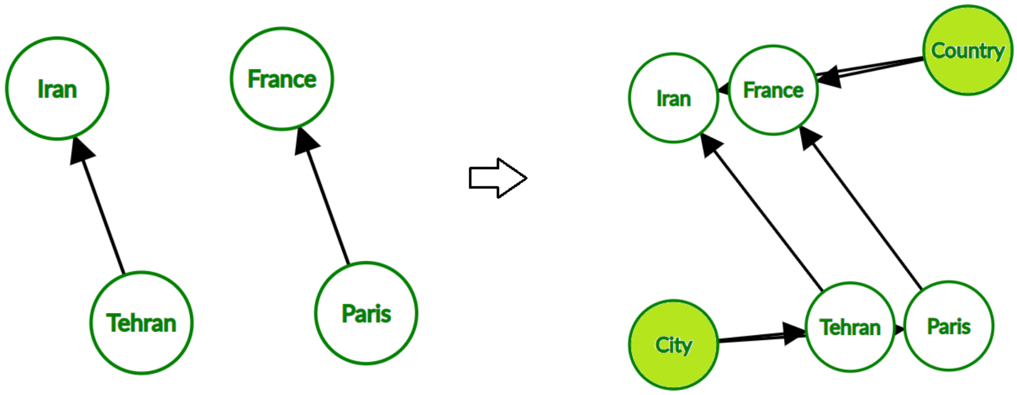

Figure 1: Simple illustration of changes in embedding space. The left diagram shows how entities and relations are embedded in the embedding space. The right side graph shows the effect of adding auxiliary nodes (Lime green nodes) to the graph, which translational models bring all countries together and cities together in vector space.

Equations 4 and 5, which are similar, bring closer the embeddings of France and Iran, which are semantically identical. Figure 1 gives an illustration of what changes KGrefiner brings for the embedding space. This closeness in evaluating Equation 2 causes the model to search between countries when asked where France's capital is.

4.1 Refinement of FB15k-237

In FB15k-237, graph relations contain information about entities. For example, the "entity → physical_entity → object → location → region → area → center → seat → capital → national_capital" is a relationship between countries and cities, and nodes on one side of relationships can be considered similar. Higher levels usually have more general information about objects in the hierarchy, and lower levels have more specific, so we extracted the last three levels of hierarchies from each relation in this graph to use this information. Then, for each sub-relation, we counted the number of repetitions in the graph training section. We removed those components with less than 100 repetitions in the graph to reduce the number of these sub-relations, and the number 100 is arbitrary. Finally, 285 sub-relations remained, which we added to the set of entities in this graph (as new nodes). We call these auxiliary nodes relation-nodes. We defined two new relations, "RelatedTo" and "HasAttribute", to connect these relation-nodes to the graph. For each triple, if the entity is the triple's head, we linked it with relation-node by "RelatedTo", and if it is the tail of the triple, we use "HasAttribute" to establish these connections. For example, to refine relation between Paris and France, (Paris,"entity → physical_entity → object → location → region → area → center → seat → capital → national_capital",France), "capital" has repetition over 100, so the following triples were added to the graph:

$$France + HasAttribute \approx capital \quad (308)$$
$$Paris + RelatedTo \approx capital \quad (309)$$

4.2 Refinement of WN18RR

To refine this graph, we use the hierarchy of entities. In Freebase, we used relationships, but relationships do not give us information about entities in Wordnet. France, for example, has a hierarchy of "existence → place → region → region → administrative region → country → France". This

Baseline	H@10	MR	MRR
TransE	45.6	347	**29.4**
TransE + Attribute	**47.6**	221	28.8
TransE + KGRefiner	47	**203**	29.1
TransD	45.3	256	**28.6**
TransD + Attribute	**48.2**	**227**	28.5
TransD + KGRefiner	43.7	**227**	24
RotatE	**47.4**	185	**29.7**
RotatE + Attribute	43.8	218	27.3
RotatE + KGRefiner	43.9	226	27.9
TransH	36.6	311	21.1
TransH + Attribute	47.7	237	28.2
TransH + KGRefiner	**48.9**	**221**	**30.2**

Table 2: Link prediction results on FB15k-237 and its refined version. Results of TransE is taken from (Nguyen et al., 2018), TransH and TransD from (Zhang et al., 2018), but for RotatE we used OpenKE (Han et al., 2018) to produce scores. For other rows, we also used OpenKE to get the scores.

Baseline	H@10	MR	MRR
TransE	50.1	3384	**22.6**
TransE + KGRefiner	**53.7**	**1125**	22.2
TransH	42.4	5875	18.6
TransH + KGRefiner	**51.4**	**1534**	**20.8**
TransD	42.8	5482	18.5
TransD + KGRefiner	**52.3**	**1348**	**21.4**
RotatE	54.7	4274	**47.3**
RotatE + KGRefiner	**57.0**	**683**	44.8

Table 3: Link prediction results on WN18RR and its refined version. Results of TransE is taken from (Nguyen et al., 2018), TransH and TransD from (Zhang et al., 2018). For other results, we used OpenKE (Han et al., 2018) to produce them.

hierarchy gives us good information about France. Except for the last level, we extract the other last three levels of entities. Among these levels, we hold those with more than an arbitrary number of 50 repetitions among entities to reduce these levels. As a result, 207 levels remained. We add these levels as new nodes to the graph training section and connect them to the entities with these levels in their hierarchy with a new type of connection. In this graph, we define a new relation and name it "HasAttribute". For example, France and Iran have a "country" in their hierarchical structure. Then, the following triples were added to the training section of the graph:

$$France + HasAttribute \approx country$$

$$Iran + HasAttribute \approx country$$

5 Exprement

5.1 Datasets

We evaluated our work on popular benchmarks: FB15k-237 and WN18RR; these datasets are respectively refined from real knowledge graphs: WordNet (Miller, 1995) and Freebase (Bollacker et al., 2008). In addition, we built two other datasets with KGRefiner: FB15k-237-Refined and WN18RR-Refined, respectively, from FB15k-237 and WN18RR. The details of the datasets are shown in Table 1.

5.2 Baselines

To demonstrate the effectiveness of our models, we compare results with the original translational models TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransD (Ji et al., 2015), and the last translational model, RotatE (Sun et al., 2019).
In addition, we used FB15k-237-Attr (Shang et al., 2019) to compare our work with other data aug-

Model	Time for single epoch
TransE (Bordes et al., 2013) [⊕]	2.8 s
TransH (Wang et al., 2014) [⊕]	5.2 s
TransD (Ji et al., 2015) [⊕]	5.2 s
RotatE (Sun et al., 2019) [⊕]	5 s
ConvE (Dettmers et al., 2018) [⊖]	279 s
ConvKB (Nguyen et al., 2018) [⊖]	40 s

Table 4: Comparison between translational technique and deep learning methods in training time. [⊕]: These models are implemented by OpenKE (Han et al., 2018) and [⊖] are produced by their original implementations.

mentation methods as base models plus attribute. For WN18RR, GrCluster (Ranganathan et al., 2020) tried to improve link prediction on wordnet by using hierarchical data using path similarity. Nevertheless, their report did not show improvement onWN18RR.

5.3 Experimental Settings

We used implementation of baselines by OpenKE (Han et al., 2018). We used an embedding dimension of 200 for all models. Also, we removed self adversarial negative sampling from TransE and RotatE to have a fair comparison. We tried {200, 500, 1000, 2000} epochs, and we picked the best epoch according to MRR on the validation set. Other hyperparameters of the models are those mentioned in OpenKE. Hyperparameters for FB15k-237 and FB15k-237-Refined and also WN18RR and WN18RR-Refined are the same.

5.4 Experimental Results

Table 2 and 3 compares the experimental results of our KGRefiner plus translational models and with previously published results. Results in bold font are the best results in the group, and the underlined results denote the best results in the column. KGRefiner with TransH obtains the highest H@10 and MRR on FB15k-237, and also KGRefiner with RotatE reached the best MR and H@10 in WN18RR.

5.5 Speed of Models

The training time of translational models is much less than deep learning approaches such as ConvE, SACN, ConvKB, etc. The complexity in scoring function and neural network layers in their architecture reduces training speed in deep learning methods. Table 4 compares the time that each model needs to be trained for one epoch on FB15k-237. We ran models on Nvidia K80. For fair comparison embedding dimension for all models is 200. These models usually need 1000 epochs, so the runtime difference between TransE and RotatE is around 35000s for FB15k-237.

6 Conclusion

In this paper, we propose KGRefiner, a novel knowledge graph refinement method that alleviates the limitations of translational models by capturing additional information in knowledge graph hierarchies. We used hierarchy components as new nodes, and by connecting these nodes to proper entities in the knowledge graph, we have a more informative graph. Our experimental results show that our KGRefiner outperforms other state-of-the-art translational models on two benchmark datasets WN18RR and FB15k-237. Furthermore, it is the first augmentation method that works with both Wordnet and Freebase, while old methods only perform only on one dataset.

In future works, we will expand our work on datasets that can be formulated on the triple structure. For example, recommender system datasets can be formed on graph schema, and KGRefiner can be applied. Additionally, KGRefiner cannot improve the accuracy of deep learning methods; therefore, another study is needed to enhance deep models by using ontological information.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, pages 1247–1250, New York, NY, USA. ACM.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014a. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 615–620.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014b. Open question answering with weakly supervised embedding models. In *Machine Learning and Knowledge Discovery in Databases*, pages 165–180, Berlin, Heidelberg.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, pages 1811–1818.

Xu Han, Shulin Cao, Lv Xin, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. Openke: An open toolkit for knowledge embedding. In *Proceedings of EMNLP*.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing (volume 1: Long papers)*, pages 687–696.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Twenty-ninth AAAI conference on artificial intelligence*.

George A. Miller. 1995. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the 47th Annual Meeting of the ACL*, pages 1003–1011.

Changsung Moon, Paul Jones, and Nagiza F Samatova. 2017. Learning entity type embeddings for knowledge graph completion. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pages 2215–2218.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *Proceedings of North American Chapter of the Association for Computational Linguistics*, pages 327–333.

Leo L Pipino, Yang W Lee, and Richard Y Wang. 2002. Data quality assessment. *Communications of the ACM*, 45(4):211–218.

Varun Ranganathan, Siddharth Suresh, Yash Mathur, Natarajan Subramanyam, and Denilson Barbosa. 2020. Grcluster: a score function to model hierarchy in knowledge graph embeddings. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pages 964–971.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end structure-aware convolutional networks for knowledge base completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3060–3067.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, AAAI'14, pages 1112–1119. AAAI Press.

Chenyan Xiong, Russell Power, and Jamie Callan. 2017. Explicit semantic ranking for academic search via knowledge graph embedding. In *Proceedings of the 26th international conference on world wide web*, pages 1271–1279.

Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 353–362, New York, NY, USA.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3065–3072.

Zhao Zhang, Fuzhen Zhuang, Meng Qu, Fen Lin, and Qing He. 2018. Knowledge graph embedding with hierarchical relation structure. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3198–3207.

Sendong Zhao, Bing Qin, Ting Liu, and Fei Wang. 2020. Biomedical knowledge graph refinement with embedding and logic rules. *arXiv preprint arXiv:2012.01031*.