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

Mohammad Javad Saeedizade[1] **Najmeh Torabian**[2] **Behrouz Minae-Bidgol**[1]
[1] School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran
[2] Department of Computer Engineering, Central Tehran Branch, Islamic Azad University, Tehran, Iran
saeedizade74@gmail.com
najmeh.torabian@gmail.com
b_minaei@iust.ac.ir

Abstract

Link Prediction is the task of predicting missing relations between knowledge graph(KG) entities. Recent work in link prediction mainly attempted to adapt a model to increase link prediction accuracy by using more layers in neural network architecture, which heavily rely on computational resources. This paper proposes the refinement of knowledge graphs to perform link prediction operations more accurately using relatively fast translational models. Translational link prediction models have significantly less complexity (faster) than deep learning approaches but are less accurate; this motivated us to improve their accuracy. Our method uses the ontologies of knowledge graphs to add information as auxiliary nodes to the graph. Then, these auxiliary nodes are connected to ordinary nodes of the KG that contain auxiliary information in their hierarchy. Our experiments show that our method can significantly increase the performance of translational link prediction methods in Hit@10, Mean Rank, and Mean Reciprocal Rank, with the same complexity as translational models.

1 Introduction

Knowledge graphs (KGs) represent a set of interconnected descriptions of entities, including objects, events, or concepts. These graphs are structures by which knowledge is captured in the form of triplets. These triplets consist of three parts: head, relation, and tail. The relation (edge) determines the type of relationship between head and tail nodes.

Despite many efforts to build KGs, they are far from completeness. One of the developing fields in completing KGs is link prediction (LP). LP tries to embed entities and relations in a small continuous vector space to predict missing links in KGs. In the last few years, deep learning approaches have significantly outperformed other methods in LP, but this accuracy came at the cost of computational complexity.

Translational LP models, such as TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransD (Ji et al., 2015), RotatE (Sun et al., 2019b), and HAKE (Zhang et al., 2020), generally use a straightforward function over head and relation vectors to predict the tail based on distance (Rossi et al., 2021) (Wang et al., 2021). One advantage of translational methods over deep learning techniques is that their score function is considerably faster (Sun et al., 2019a). Since these models are less complex and more efficient, we tried to improve only these translational methods in this work.

Ontologies are concepts or properties to describe an object [1]. Wordnet contains hierarchical ontology only for its entities. Some work tried to use ontology components of Wordnet to boost LP models. For example, GrCluster (Ranganathan et al., 2020) treated ontology components as paths. It defined path similarity over entities in Wordnet and slightly improved LP accuracy. Nonetheless, GrCluster only improved WNNH and WN18, which are not standard LP datasets (Dettmers et al., 2018). Additionally, this work is limited to Wordnet.

Freebase (Bollacker et al., 2008) does not have any hierarchical path for its entity. On the other hand, its relations have a path hierarchy to explain edges. SACN (Shang et al., 2019) exploited additional information of FB15k-237 as auxiliary nodes and created FB15k-237-Attr. Nevertheless, it added numerous nodes to the KG, which makes the method for creating FB15k-237-Attr inefficient for more extensive graphs. Likewise, this method can only be applied to Freebase.

Translational LP models, such as TransE, RotatE, or TransD, when trying to learn the relation between Paris and France, neglect that Paris is a city and France is a country. We introduce ontology components as auxiliary nodes. These auxiliary nodes are connected to related entities that

[1] https:en.wikipedia.orgwikiOntology_(information_science)

have these components in their hierarchy. For example, we added an extra node "country" to KG and connected it to all the countries in the KG. Our contributions are as follows:

Firstly, we presented a method for refining KGs that have ontology. Our approach adds auxiliary nodes and embeds similar nodes closer in the embedding space, which increases the accuracy of translational link prediction with the same time and space complexity of translational models. **Secondly**, we used state-of-the-art translational models to evaluate our method on two FB15k-237 and WN18RR. The results showed that accuracy in link prediction was significantly increased on H@10, MRR, and MR, especially on WN18RR.

2 Related Work

We divided related work into four categories.

First, translational models, such as TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransD (Ji et al., 2015), HAKE (Zhang et al., 2020), are distance-based algorithms that use a straightforward operation over head and relation (mainly summation and/or a projection into a secondary space) to measure the distance to the tail entity. Some work has been introduced over these fast translational models to improve their performance by using hierarchical information. TKRL (Xie et al., 2016) used components of hierarchical structure as a transition to transform KG nodes into secondary space and then performed LP. GrCluster (Ranganathan et al., 2020) used path similarity over entities in Wordnet and slightly improved link prediction accuracy. SACN (Shang et al., 2019) proposed FB15k-237_Attr that has external resources as triplets (new nodes and edges) to improve the result.

GrCluster could not improve the WN18RR, and it is limited to KGs that have ontology for their entities. SACN improved FB15k-237 by creating FB15k-237_Attr, but it added many nodes and edges. Nonetheless, the SACN attribute creator could not be applied to WN18RR. TransC (Lv et al., 2018) brought similar entities closer in the embedding space and improved LP in YAGO, but experiment results show no improvements on Wordnet or Freebase. Our work is similar to this category; It is fast and uses translational models as a core. We pushed the limitation of TransC to have a better LP result on Freebase and Wordnet.

Second, mostly deep models adapt an architecture and rarely use anthologies in their main model. For example, ConvE (Dettmers et al., 2018) used 2D convolution, BERT-ResNet (Lovelace et al., 2021) and KG-BERT (Yao et al., 2019) employed BERT, SACN (Shang et al., 2019) utilized WGCN in its architecture. These models are more accurate but computationally costly.

Thirdly, KG refinement is a sub-field of KG enhancement. Refinement can be done by either adding information to the graph or removing incorrect data (Paulheim, 2017). BioKG (Zhao et al., 2020) worked on medical KGs and has tried to provide a method for removing the inaccurate information in these graphs. In this work, like SACN, we added auxiliary nodes to KGs. These nodes are extracted from ontology hierarchy levels of nodes and edges of KGs.

Lastly, some works introduced similarities over entities or relations. For example, HRS (Zhang et al., 2018) presented relation-cluster and subrelations in the scoring function of translational models. It created sub-relations and relation-clusters based on clustering results of TransE relations; however, it cannot utilize ontology nor improve WN18RR results. For entity similarity, ETE (Moon et al., 2017) considered that if two entities are embedded closely in the embedding space, they are similar and assigned classes to entities based on closeness. Unlike ETE, our hypothesis is that if two entities use the same relation type in the graph or have common elements in their hierarchies, they are related. We exploited these affiliations (share hierarchical components) by connecting ordinary nodes to their auxiliary nodes if a node has the auxiliary node in its ontology components.

The main distinctions between our work and related work are: First, our method works with any KG with ontology, and it does not matter if it has the hierarchical ontology for nodes or edges. Second, it uses translational models; therefore, it has high speed and less time to train these models (see Table 3).

3 KGRefiner

In this work, we propose a method that uses ontology as an auxiliary node, which refines the KG and increases LP accuracy. These auxiliary nodes can be obtained from the edges of KG or its nodes. For example, in FB15k-237, we do this refinement by using hierarchies of relations, and in WN18RR, we use hierarchies of entities. We add repetitive components of hierarchies to KGs as new (auxiliary) nodes. Then, we introduce a few new relations to connect these auxiliary nodes to other KG nodes.

These auxiliary nodes operate like a magnet for similar entities; They drag similar entities (those entities that share ontology components) together in the embedding space. This closeness of similar entities causes the translational models to prioritize their search for a specific place in the embedding space (e.g., searching between countries when asked what country's capital city is Paris, in evaluation). The rest of this section is dedicated to the proof of this assertion.

Translational link prediction methods, such as TransE, create transition property in their embeddings. For example, in TransE, embeddings are made as $\vec{e_s} + \vec{r} \approx \vec{e_o}$. This means the tail entity should be close to the sum of head and relation in embedding space. Let us consider n entities share an ontology component O in their hierarchy. If we add O to the KG and connect the O to those n entities, the following optimization will happen in TransE:

$$\begin{aligned}\vec{E_1} + \vec{RelatedTo} &\approx \vec{O}\\ \vec{E_2} + \vec{RelatedTo} &\approx \vec{O}\\ &\dots\\ \vec{E_n} + \vec{RelatedTo} &\approx \vec{O}\end{aligned}$$

The loss function minimizes the distance between two sides of equations:

$$\begin{aligned}Loss = {} & ||\vec{E_1}+\vec{RelatedTo} - \vec{O}||+\\ & ||\vec{E_2}+\vec{RelatedTo} - \vec{O}||+\\ & \dots\\ & ||\vec{E_n}+\vec{RelatedTo} - \vec{O}||\end{aligned}$$

In the implementation, they are optimized batch-wised. Also, assume it uses the L1 norm as a distance measure. Therefore, the batch loss will be:

$$Loss = \sum_{n=1}^{n} Distance(\vec{E_i} + \vec{RelatedTo}, \vec{O})$$

$$= \sum_{n=1}^{n} Distance(\vec{E_i}, \vec{O} - \vec{RelatedTo})$$

$$= \sum_{n=1}^{n} ||\vec{E_i}, \vec{O} - \vec{RelatedTo}||_1$$

Since $(\vec{O} - \vec{RelatedTo})$ can be considered constant, all $\vec{E_i}$ will be dragged to where $(\vec{O} - \vec{RelatedTo})$ is located in the embedding space. For example, if we connect all KG countries to an ontology node "country", then all countries will be embedded closer.

3.1 Refinement of FB15k-237

In FB15k-237, graph relations reflect information about entities. For example, in (Paris, national_capital, France), national_capital has hierarchy of "entity $\rightarrow$ physical_entity $\rightarrow$ object $\rightarrow$ location $\rightarrow$ region $\rightarrow$ area $\rightarrow$ center $\rightarrow$ seat $\rightarrow$ capital $\rightarrow$ national_capital". This hierarchy is a relationship between countries and their capitals, and nodes on one side of relationships (e.g. left side of triplet) can be considered similar (e.g. they are countries). Moreover, higher hierarchy levels usually have more abstract information about objects, but the lower ones are more specific. Therefore, we extracted the last three levels of hierarchies from each relation in this KG to use hierarchy components. Then, for each sub-relation (component), we counted the number of their repetitions in the KG training section triplets. Then, we removed those components with less than 100 repetitions to reduce the number of these components; the number 100 is arbitrary. Finally, 285 sub-relations remained, and we added them to the set of entities in this KG (as auxiliary nodes). We defined two new relations, "RelatedTo" and "HasAttribute", to connect these relation-nodes (auxiliary nodes) to the KG entities. For each triplet, if the entity is the triplet's head, we link it to the auxiliary node by "RelatedTo", and if it is the tail of the triplet, we use "HasAttribute" to establish these connections. For example, to

Algorithm 1: Refinement of FB15k-237

```
Input ($TrainTriplets, Hierarchies, MinRep. = 100$)
    $Hierarchies \leftarrow$ LastLevels($Hierarchies, 3$)

    $Hierarchies \leftarrow$ Repetitives($Hierarchies, MinRep$)

    $NewEdges = []$
    for all $(h, r, t)$ in $TrainTriplets$ do
        for all $H$ in $Hierarchies$ do
            $NewEdges \leftarrow NewEdges + (h, HasAttribute, H)$

            $NewEdges \leftarrow NewEdges + (t, RelatedTo, H)$
    return $TrainTriplets + NewEdges$
```

refine relation between Paris and France, (Paris, entity $\rightarrow$ physical_entity $\rightarrow$ object $\rightarrow$ location $\rightarrow$ region $\rightarrow$ area $\rightarrow$ center $\rightarrow$ seat $\rightarrow$ capital $\rightarrow$ national_capital, France), “capital” has repetition over 100, so the following triplets were added to the graph:

$$(France, HasAttribute, capital)$$
$$(Paris, RelatedTo, capital)$$

3.2 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 $\rightarrow$ place $\rightarrow$ region $\rightarrow$ region $\rightarrow$ administrative region $\rightarrow$ country”. This hierarchy gives us good information about France. We extract the last three levels of entities. Among these levels, we hold those with more than an arbitrary number of 50 repetitions among entities to reduce the number of auxiliary nodes. As a result, 207 levels remained. We add these levels as new nodes to the KG training section and connect them to entities that have these components in their hierarchy with a new type of connection “HasAttribute”. For example, France and Iran have a “country” in their hierarchical structure. Then, the following triplets were added to the training section of the graph:

$$(France, HasAttribute, country)$$
$$(Iran, HasAttribute, country)$$

Algorithm 2: Refinement of WN18RR

```
Input
  ($TrainTriplets, Hierarchies, Entities, MinRep. = 50$)
    $Hierarchies \leftarrow$ LastLevels($Hierarchies, 3$)

    $Hierarchies \leftarrow$
      Repetitives($Hierarchies, MinRep$)

    $NewEdges = []$
    for all $e$ in $Entities$ do
        for all $H$ in $Hierarchies$ do
            if $H$ IsComponentOf $e$ then
                $NewEdges \leftarrow NewEdges +$
                $(e, HasAttribute, H)$
    return $TrainTriplets + NewEdges$
```

3.3 New Relations

We introduce new edge types to connect auxiliary nodes to the KG to make them distinguishable from original relation types. Since in WN18RR it is only one relation is needed, we introduce "HasAttribute" to say this node has this ontology attribute in its hierarchy. However, in FB15k-237, only edges have ontology components. Therefore, we need to know on which side of the edge an entity is located (head or tail). Therefore, we introduced two new relations: "HasAttribute" and "RelatedTo".

4 Exprement

4.1 Datasets

We evaluated our work on popular benchmarks: FB15k-237 and WN18RR. In addition, we built two other datasets with KGRefiner: FB15k-237-Refined and WN18RR-Refined from those datasets. The details of the datasets are available in appendix in Table 4.

4.2 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), RotatE (Sun et al., 2019b), and HAKE (Zhang et al., 2020), with fair setting (see Section 4.4 and Appendix A). In addition, we used FB15k-237-Attr (Shang et al., 2019) to compare our work with other data augmentation methods as base models plus attributes.

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 in WN18RR.

4.3 Experimental Results

Table 1 and 2 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.

In tables, results of TransE is taken from (Nguyen et al., 2018), TransH from (Zhang et al., 2018). For other rows, we used OpenKE (Han et al., 2018) and original HAKE implementation 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**
HAKE	52.2	4433	**40.0**
HAKE + KGRefiner	**53.8**	**2125**	25.0
RotatE	54.7	4274	**47.3**
RotatE + KGRefiner	**57.0**	**683**	44.8

Table 1: Link prediction results on WN18RR and its refined version.

Baseline	H@10	MR	MRR
TransE	45.6	347	**29.4**
TransE + Attribute	**47.6**	221	28.8
TransE + KGRefiner	47	**203**	29.1
HAKE	**40.8**	282	**23.8**
HAKE + Attribute	38.4	287	21.7
HAKE + KGRefiner	39.0	**267**	21.4
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. The "+ Attribute" is the refined version produced by (Shang et al., 2019)

4.4 Speed of Models

The training time of translational models is much less than deep learning approaches such as ConvE, SACN, ConvKB, etc. The complexity of scoring function and neural network layers in their architecture reduces training speed in deep learning methods. Table 3 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. It can be observed that the runtime difference between our best result with KGRefiner (TransH + KGRefiner) and BERT-ResNet (Lovelace et al., 2021) for a small dataset FB15k-237 is around 9.6×10^5s. In other words, our method is 100 times faster. In terms of their accuracy (H@10, MRR, MR), BERT-ResNet scores are (0.514, 0.346, 186) but TransH + KGRefiner are (0.489, 0.302, 221). The scores are slightly lower, but speed is uncomparable.
Apart from that, according to table 4, KGRefiner adds triplets to the training section of these KGs. Therefore, it only increases the training time of WN18RR and FB15k-237 by a factor of 2.65 and 2.02, respectively. It does not increase other measurements' complexity because it adds few nodes to the KGs. Consequently, the training cost of the translational models with KGRefiner is still much cheaper than deep learning techniques.

Model	Time to train	Time to train with KGRefiner
TransE [$\oplus$]	2.8×10^2 s	5.6×10^2 s
TransH [$\oplus$]	5.2×10^2s	1×10^3s
TransD [$\oplus$]	5.2×10^2s	1×10^3s
RotatE [$\oplus$]	5×10^2s	1×10^3s
HAKE [$\oplus$]	1.5×10^4s	3×10^4s
ConvE [$\ominus$]	2.7×10^5s	-
ConvKB [$\ominus$]	4×10^4s	-
BERT-ResNet [$\ominus$] (Lovelace et al., 2021)	9.7×10^4s	-

Table 3: Comparison between translational technique and deep learning methods in training time on the small-standard Freebase sub-graph (FB15k-237) . [$\oplus$]: These models are implemented by OpenKE (Han et al., 2018) and [$\ominus$] are produced by their original implementations.

5 Conclusion and Future work

In this work, we propose KGRefiner, a KG refinement method that alleviates the limitations of translational models by capturing additional information in knowledge graph hierarchies. We used hierarchy components as auxiliary nodes. Refined KG comes by connecting these auxiliary nodes to proper entities. Our empirical results show that our KGRefiner outperforms other state-of-the-art translational models and data augmentation methods on WN18RR. Some models' performance improved on FB15k-237 but was not as good as WN18RR. Furthermore, it is the first augmentation method that works with both Wordnet and Freebase, while old methods only perform only on one dataset.

In our work, we had to manually determine the depth cut of hierarchy and minimum repetition for ontology components extraction. In future works, we will automate these two elements, so the model determines each component. 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, 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.

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.

Justin Lovelace, Denis Newman-Griffis, Shikhar Vashishth, Jill Fain Lehman, and Carolyn Rosé. 2021. Robust knowledge graph completion with stacked convolutions and a student re-ranking network. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1016–1029, Online. Association for Computational Linguistics.

Xin Lv, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2018. Differentiating concepts and instances for knowledge graph embedding. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1971–1979.

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.

Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic web*, 8(3):489–508.

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.

Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Merialdo. 2021. Knowledge graph embedding for link prediction: A comparative analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(2):1–49.

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.

Zequn Sun, Jiacheng Huang, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019a. Transedge: Translating relation-contextualized embeddings for knowledge graphs. In *International Semantic Web Conference*, pages 612–629. Springer.

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

Meihong Wang, Linling Qiu, and Xiaoli Wang. 2021. A survey on knowledge graph embeddings for link prediction. *Symmetry*, 13(3):485.

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.

Ruobing Xie, Zhiyuan Liu, Maosong Sun, et al. 2016. Representation learning of knowledge graphs with hierarchical types. In *IJCAI*, volume 2016, pages 2965–2971.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*.

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*.

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	3134	3134	20466

Table 4: 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.

A Hyperparameter Settings

We employed the implementation of baselines by OpenKE (Han et al., 2018), and HAKE (Zhang et al., 2020) to produce the result.

To have a fair comparison between translational models, we used an embedding dimension of 200 for all models (to produce the same result as in their paper, some models need more than 1000 dimensions for entity embedding). Also, we removed self adversarial negative sampling from TransE, RotatE, and HAKE and replaced it with typical negative sampling. Moreover, we tried {200, 500, 1000, 2000} epochs, and we picked the best one according to MRR on the validation set for final comparison. Other hyperparameters of the models are those mentioned in OpenKE and HAKE. Hyperparameters for FB15k-237 and FB15k-237-Refined and also WN18RR and WN18RR-Refined are the same. Interestingly, HAKE heavily relied on 1000 embedding dimensions to reproduce the result on its paper.

B Limitations

KGRefiner needs a KG that has ontology for either its nodes or edges. Therefore, in other developing KGs, KGRefiner cannot be applied. In addition, since it brings similar entities closer, this can only improve distance-based models (translational).