Relational Graph Neural Network with Hierarchical Attention for Knowledge Graph Completion

Zhao Zhang[1,2], **Fuzhen Zhuang**[1,2,*], **Hengshu Zhu**[3], **Zhiping Shi**[4], **Hui Xiong**[3,5], **Qing He**[1,2]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing 100190, China
[2]University of Chinese Academy of Sciences, Beijing 100049, China
[3]Baidu Talent Intelligence Center, Baidu Inc., Beijing, China
[4]Capital Normal University, Beijing 100048, China
[5]Business Intelligence Lab, Baidu Inc., Beijing, China
{zhangzhao2017, zhuangfuzhen, heqing}@ict.ac.cn, zhuhengshu@baidu.com, shizp@cnu.edu.cn, xionghui@gmail.com

Abstract

The rapid proliferation of knowledge graphs (KGs) has changed the paradigm for various AI-related applications. Despite their large sizes, modern KGs are far from complete and comprehensive. This has motivated the research in knowledge graph completion (KGC), which aims to infer missing values in incomplete knowledge triples. However, most existing KGC models treat the triples in KGs independently without leveraging the inherent and valuable information from the local neighborhood surrounding an entity. To this end, we propose a Relational Graph neural network with Hierarchical ATtention (RGHAT) for the KGC task. The proposed model is equipped with a two-level attention mechanism: (i) the first level is the relation-level attention, which is inspired by the intuition that different relations have different weights for indicating an entity; (ii) the second level is the entity-level attention, which enables our model to highlight the importance of different neighboring entities under the same relation. The hierarchical attention mechanism makes our model more effective to utilize the neighborhood information of an entity. Finally, we extensively validate the superiority of RGHAT against various state-of-the-art baselines.

Introduction

Nowadays, large-scale knowledge graphs (KGs) have become one of the most important resources for enhancing AI-related applications, such as information retrieval (Dalton, Dietz, and Allan 2014), question answering (Ferrucci et al. 2010), and information extraction (Mintz et al. 2009). Indeed, KGs can be represented as multi-relational directed graphs composed of entities as nodes and relations as edges. The information of real-world entities and relations is modeled in the form of knowledge triples, which are denoted as (h, r, t), where h and t correspond to the head and tail entities and r denotes the relation between them, e.g., (*Paris*, *capital_of*, *France*). Despite the success and popularity of modern KGs such as Freebase (Bollacker et al. 2008), Yago (Suchanek, Kasneci, and Weikum 2007), Gene Ontology (Ashburner et al. 2000) and NELL (Carlson et al. 2010), their coverage is still far from complete and comprehensive, which motivates the research in knowledge graph completion (KGC), i.e., to predict the missing values in incomplete knowledge triples. More formally, the goal of KGC is to predict either the head entity in a given query ($?$, r, t) or the tail entity in a given query (h, r, $?$).

In literature, state-of-the-art KGC models usually based on knowledge graph embedding (Bordes et al. 2013; Wang et al. 2014; Yang et al. 2015; Schlichtkrull et al. 2018). Specifically, these models first learn the latent low-dimensional representations of entities and relations in the KG, and then predict the missing values in incomplete knowledge triples based on linear or neural network models with corresponding representations. However, most of these models treat the triples in KGs independently, and fail to pay attention to the local neighborhood of an entity, which may also contain plenty of valuable and inherent information.

In this paper, we aim to take full advantage of the local neighborhood information of each entity for enhancing the KGC task. Specifically, we treat the neighborhood of an entity as a hierarchical structure. Figure 1 shows an example of an entity and its neighborhood. From the left part of Figure 1, it can be seen the central entity e_1 is surrounded by three neighboring relations r_1, r_2 and r_3, and each relation links e_1 to one or more neighboring entities. We find the local neighborhood of e_1 can be viewed as a hierarchical structure, which is shown in the right part of Figure 1, where $N_{h,r}$ denotes the set of neighboring entities of entity h under relation r. The intuition underlying our model is that (i) not all neighboring relations are equally relevant for representing the central entity, and (ii) not all neighboring entities in each $N_{h,r}$ are equally important in indicating the central entity.

Under the above observation, here we propose a novel neighborhood-aware model, named Relational Graph neural network with Hierarchical ATtention (RGHAT) for KGC. In RGHAT, we design a novel hierarchical attention mechanism to compute different weights for different neighboring relations and entities. Specifically, inspired by that the importance of different relations differ greatly in indicating an entity, RGHAT first computes the weights for different neighboring relations, which is the first-level attention. Next, RGHAT computes the attention scores for different neighboring entities under each relation, which is the

*Corresponding author: Fuzhen Zhuang

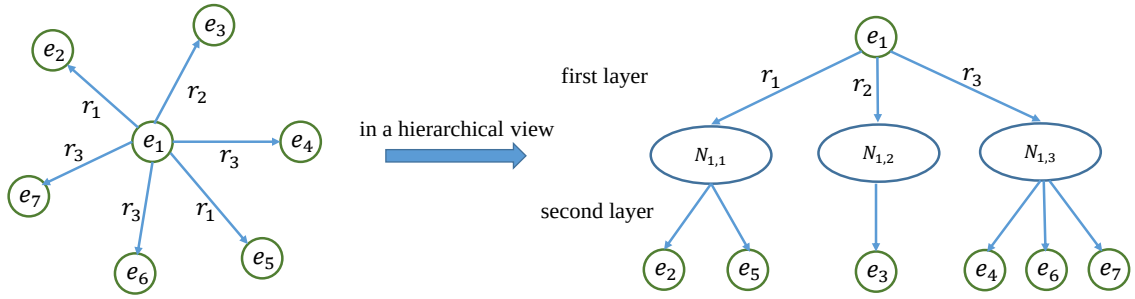

Figure 1: From star-graph to a hierarchical structure: an example of local neighborhood. $N_{h,r}$ denotes the set of neighboring entities of entity h under relation r.

second-level attention. Finally, each entity aggregates information and gets updated from its neighborhood based on the hierarchical attentions.

In particular, the advantages of our RGHAT model can be summarized as follows.

- First, different from most existing KGC models, our model explicitly takes advantage of the local neighborhood information of each entity.
- Second, the hierarchical structure can be viewed as an integration of local neighboring information, and integrating related information into groups has already been verified to be beneficial to provide more information and yield better results in machine learning problems (Yuan and Lin 2006; Yang et al. 2016).
- Third, in RGHAT, the hierarchical attention mechanism provides a fine-grained learning process for the model, which increases the interpretability of our model. Moreover, under this setting, the weights of neighboring triples with the same relations can be trained in a collective way, making the results of our model more stable and more consistent with human intuition.

Finally, extensive experiments on a number of popular benchmark datasets clearly validate the effectiveness of RGHAT against various state-of-the-art baselines.

Related Work

Recent years have witnessed increasing interest in the KGC problem. Among all the KGC methods, the most successful ones learn distributed representations for entities and relations in KGs, and predict missing values in incomplete knowledge triples using linear or neural network based operations. These models roughly fall into three categories: (i) Translation-based models, which view relations as translations from a head entity to a tail entity. TransE (Bordes et al. 2013) is one of the most widely used KGC models, which views a relation as a translation from a head entity to a tail entity on the same low-dimensional hyperplane, i.e., $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ when (h, r, t) holds. TransH (Wang et al. 2014) introduces a mechanism of projecting entities into relation-specific hyperplanes that enables different roles of an entity in different relations. TransR (Lin et al. 2015b) embeds entities and relations into separate entity space and relation-specific spaces. (ii) Tensor factorization based models, which assume the score of a triple can be factorized into several tensors. RESCAL (Nickel, Tresp, and Kriegel 2011) utilizes the score function $f(h, r, t) = \mathbf{h}\mathbf{M}_r\mathbf{t}$ to compute the scores of knowledge triples, and assumes that positive triples have higher scores than negative ones. DistMult (Yang et al. 2015) extends RESCAL and sets the relation-specific matrix $\mathbf{M}_r$ to be diagonal. ComplEx (Trouillon et al. 2016) embeds entities and relations into complex vectors instead of real-valued ones. (iii) Neural network based models, which utilize deep neural networks to embed KGs. ConvE uses a convolutional neural network based structure, while R-GCN (Schlichtkrull et al. 2018) and Nathani's (Nathani et al. 2019) use graph neural network (GNN) based structures to model KGs.

Though the number of KGC models is large, very few of them utilize the local neighborhood information of an entity. García-Durán et al. (2015) and Lin et al. (2015a) leveraged path information to embed entities and relations. Schlichtkrull et al. (2018) extended the widely used graph convolutional network to relational graphs, and achieved promising results with the information from local neighborhood. Bansal et al. (2019) and Nathani et al. (2019) further distinguished the weight of neighboring nodes, and yielded state-of-the-art performance for KGC. Along this line, we propose RGHAT, which views the local neighborhood of an entity as a hierarchical structure, and effectively compute proper attention weights for neighboring entities and relations.

Integrating information into groups or hierarchies has shown to be beneficial for machine learning problems. Yuan et al. (2006) extended lasso to group lasso by selecting feature groups instead of individual features. Evgeniou (2004) proposed regularized multi-task learning, in which similar tasks are clustered into groups, and variables of tasks from the same group share information with each other. Yang et al. (2016) regarded text as word-level and sentence-level structures, and proposed hierarchical attention networks for document classification. In this paper, we propose to treat the neighboring entities under the same relation as a group, and aggregate information from local neighborhood in a hierarchical way.

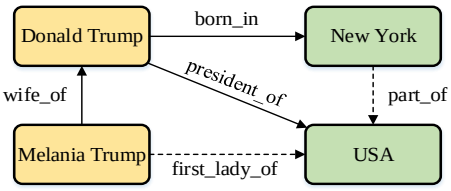

Figure 2: Subgraph of a KG containing existing triples (solid lines) and the inferred ones (dashed lines).

Methodology

In this section, we provide the technical details of the proposed RGHAT model. First, we introduce the overall framework of the proposed model. Then detailed descriptions of each step are given. Finally, we provide the loss function and training details.

Overview

A KG is viewed as a graph $\mathcal{G} = \{(h, r, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ and $\mathcal{R}$ are the entity (node) set and relation (edge) set respectively. Given a KG, we can infer missing links based on existing triples in the KG. Figure 2 provides a subset of a KG. In Figure 2, missing links (represented in dashed lines) such as (*New York, part_of, USA*) can be inferred based on existing triples (represented in solid lines) including (*Donald Trump, born_in, New York*) and (*Donald Trump, president_of, USA*).

The proposed model RGHAT follows an encoder-decoder framework. Figure 3 shows the framework of a single-layer RGHAT model. To get multi-layer RGHAT, we just need to stack the encoder into multiple layers. The encoder first views the local neighborhood of an entity as a hierarchical structure, and computes the relation-level attention and entity-level attention for the neighborhood. Next, the two attention scores are combined into a triple-level attention score, and the final score is fed forward to the information aggregator, which can effectively aggregate local neighborhood information into the central entity. Finally, the encoder outputs the entity embeddings to the decoder. The decoder is a KGC model, which can be substitute by a number of existing KGC models. This setting guarantees the flexibility and extendibility of our model. In this paper, we choose ConvE (Dettmers et al. 2018) as our decoder. In the following, we will introduce the details of the encoder and decoder.

Encoder

In this section, we give detailed descriptions of the hierarchical attention mechanism and the information aggregator.

Relation-level Attention The relation-level attention is inspired by the fact that the weights of different relations differ greatly in indicating an entity. For example, intuitively, the relation *has_players* is more indicative than the relation *based_in_city* when representing the basketball team *Los Angeles Lakers*, since the players of a team can uniquely identify the team, while there may be more than one team that are based in the same city.

The neighboring entities and relations can be represented as a number of (h, r, t) triples. In this paper, for an entity e_1, we transform the triple (e_2, r, e_1) into (e_1, r^{-1}, e_2). In this way, each entity can always act as the head entity of the neighborhood triples. And we only need to aggregate information from tail entities to the head entity.

For entity h, the relation-level attention indicates the weight of each relation when representing the entity, which is defined as

$$\mathbf{a}_{h,r} = \mathbf{W}_1 \left[\mathbf{h} \,\|\, \mathbf{v}_r \right], \tag{1}$$

$$\alpha_{h,r} = \text{softmax}_r(\mathbf{a}_{h,r}) = \frac{\exp(\sigma(\mathbf{p} \cdot \mathbf{a}_{h,r}))}{\sum_{r' \in \mathcal{N}_h} \exp(\sigma(\mathbf{p} \cdot \mathbf{a}_{h,r'}))}, \tag{2}$$

where $\|$ represents the concatenation operation, $\mathbf{h} \in \mathbb{R}^d$ is the embedding of the entity h, and d is the embedding size. $\mathbf{W}_1 \in \mathbb{R}^{d \times 2d}$, $\mathbf{v}_r \in \mathbb{R}^d$ and $\mathbf{p} \in \mathbb{R}^d$ are training parameters, in which $\mathbf{v}_r$ is a relation-specific parameter representing the characteristics of a relation. $\mathcal{N}_h$ represents the neighboring relations of entity h. σ represents the LeakyReLU function with negative input slope as 0.2. After the above calculation, we get the relation-level attention score $\alpha_{h,r}$, which represents the weight of relation r when representing entity h.

Entity-level Attention The entity-level attention is inspired by the intuition that the weights of neighboring entities under the same relation may also be different. For instance, the relation *has_players* links *Los Angeles Lakers* to different players, among all these players, top stars may be more indicative than other players.

The proposed model first views the neighboring entities under the same relation as a group, then computes the entity-level attention as follows,

$$\mathbf{b}_{h,r,t} = \mathbf{W}_2 \left[\mathbf{a}_{h,r} \,\|\, \mathbf{t} \right], \tag{3}$$

$$\beta_{r,t} = \text{softmax}_t(\mathbf{b}_{h,r,t}) = \frac{\exp(\sigma(\mathbf{q} \cdot \mathbf{b}_{h,r,t}))}{\sum_{t' \in \mathcal{N}_{h,r}} \exp(\sigma(\mathbf{q} \cdot \mathbf{b}_{h,r,t'}))}, \tag{4}$$

where $\mathbf{t} \in \mathbb{R}^d$ is the embedding of the entity t, and t is a tail entity under relation r. $\mathcal{N}_{h,r}$ represents the tail entities of entity h under relation r. $\mathbf{W}_2 \in \mathbb{R}^{d \times 2d}$ and $\mathbf{p} \in \mathbb{R}^d$ are the training parameters. $\mathbf{b}_{h,r,t}$ can be regarded as the representation of the neighboring triple (h, r, t). $\beta_{r,t}$ is the entity-level attention score, which denotes the weight of entity t among all the tail entities under relation r when representing h.

After obtaining the relation-level attention and the entity-level attention, the two scores are further combined into a triple-level attention score, which is computed as follows,

$$\mu_{h,r,t} = \alpha_{h,r} \cdot \beta_{r,t}, \tag{5}$$

where the triple-level attention score $\mu_{h,r,t}$ represents the weight of the triple (h, r, t) when representing h. The hierarchical attention mechanism provides a fine-grained learning process for the attention score, which increases the interpretability of the model. Moreover, it is worth noting that the relation-level attention $\alpha_{h,r}$ is explicitly shared by all the neighboring triples under relation r, which facilitates the knowledge sharing among these triples, and enables the weights of neighboring triples with relation r to be trained in a collective way.

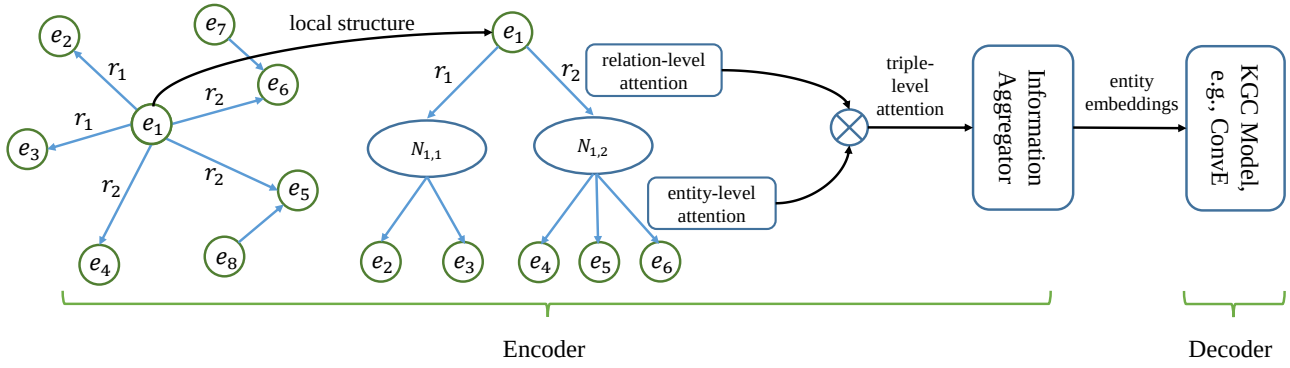

Figure 3: The overall framework of a single-layer RGHAT model.

Information Aggregator The information aggregator aggregates information from the local neighborhood to the central entity, and gets the neighborhood-based representation for entity h, which is computed as

$$\hat{\mathbf{h}} = \sum_{r \in \mathcal{N}_h} \sum_{t \in \mathcal{N}_{h,r}} \mu_{h,r,t} \mathbf{b}_{h,r,t}. \tag{6}$$

Though $\hat{\mathbf{h}}$ incorporates the information from its local neighborhood, it lacks valuable information from itself. To this end, we further combine the neighborhood-based representation $\hat{\mathbf{h}}$ with the input representation $\mathbf{h}$, and get the output representation $\mathbf{h}'$. In this paper, we design three combination ways for $\hat{\mathbf{h}}$ and $\mathbf{h}$,

- Additive combination:

$$\mathbf{h}' = \sigma(\mathbf{W}_3(\mathbf{h} + \hat{\mathbf{h}})), \tag{7}$$

  where $\mathbf{W}_3 \in \mathbb{R}^{d' \times d}$ is the training parameter, which projects the combination of $\hat{\mathbf{h}}$ and $\mathbf{h}$ to the output space, and d' is the size of the output space.

- Multiplicative combination:

$$\mathbf{h}' = \sigma(\mathbf{W}_4(\mathbf{h} \odot \hat{\mathbf{h}})), \tag{8}$$

  where $\odot$ denotes the Hadamard (element-wise) multiplication, and $\mathbf{W}_4 \in \mathbb{R}^{d' \times d}$ is the training parameter.

- Bi-interaction combination:

$$\mathbf{h}' = \frac{1}{2}\left(\sigma(\mathbf{W}_3(\mathbf{h} + \hat{\mathbf{h}})) + \sigma(\mathbf{W}_4(\mathbf{h} \odot \hat{\mathbf{h}}))\right), \tag{9}$$

  which is a combination of the above two methods.

As suggested by the graph attention network (Veličković et al. 2018), we also utilize the multi-head attention (Vaswani et al. 2017) to stabilize the learning process and encapsulate more information about the neighborhood. Specifically, K dependent attention mechanisms calculate the embeddings, which are then concatenated, resulting in the following representation

$$\mathbf{h}' = \Big\Vert_{k=1}^{K} \mathbf{h}'_k, \tag{10}$$

where $\mathbf{h}'_k$ is the output representation of the k-th head. In the final layer of the encoder, we average the embeddings from multiple heads instead of concatenating them, which is shown as

$$\mathbf{h}' = \frac{1}{K} \sum_{k=1}^{K} \mathbf{h}'_k. \tag{11}$$

A single-layer encoder aggregates information from 1-hop neighbors to the central entity in 1 training iteration. With the numbers of layers and iterations increase, our model can effectively aggregate information from multi-hop neighbors, which provide valuable information for representing the central entity. Finally, the encoder outputs the new entity embedding $\mathbf{h}'$ to the decoder.

Decoder

The decoder can be substituted by a number of existing KGC models. Particularly, we use ConvE (Dettmers et al. 2018) as the decoder in this paper. It is worth noting that we also tried different models as the decoder, but found that using ConvE achieved the best performance. ConvE models the interactions between input entities and relations by convolutional and fully-connected layers. Given (h, r, t) triples, ConvE first reshapes the embedding of h and r into 2D tensors, then computes the scores of knowledge triples based on the reshaped tensors. The score function of ConvE is

$$f(h, r, t) = \text{ReLU}\left(\text{vec}(\text{ReLU}([\bar{\mathbf{h}}; \bar{\mathbf{r}}] * \omega))\mathbf{Q}\right)\mathbf{t}, \tag{12}$$

where $\bar{\mathbf{h}}$ and $\bar{\mathbf{r}}$ are 2D reshapings of $\mathbf{h}$ and $\mathbf{r}$: if $\mathbf{h}, \mathbf{r} \in \mathbb{R}^{d'}$, then $\bar{\mathbf{h}}, \bar{\mathbf{r}} \in \mathbb{R}^{d_1 \times d_2}$, where $d' = d_1 d_2$. ω denotes a set of filters, and $*$ denotes the convolution operator. $\text{vec}(\cdot)$ is a vectorization function, and $\mathbf{Q}$ is the weight matrix. ConvE assumes that positive triples have higher scores than negative ones.

The loss function of the proposed RGHAT model is defined as follows,

$$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{T}_t} -\frac{1}{N} \sum_{i=1}^{N} (y_{(h,r,t_i)} \cdot \log(g(f(h, r, t_i))) + (1 - y_{(h,r,t_i)}) \log(1 - g(f(h, r, t_i)))), \tag{13}$$

where $y_{(h,r,t_i)}$ is the label (1 or 0) of the triple (h, r, t_i). N denotes the number of candidates for the tail entity, and g is the sigmoid function. The learning process of the proposed model is carried out using the Adam optimizer (Kingma and Ba 2014).

Experiments

In this section, we evaluate the performance of RGHAT on the task of knowledge graph completion.

Datasets

We evaluate the RGHAT model on four popular benchmark datasets FB15k (Bordes et al. 2013), WN18 (Bordes et al. 2013), FB15k-237 (Toutanova and Chen 2015) and WN18RR (Dettmers et al. 2018). FB15k is a subset of the widely used KG Freebase (Bollacker et al. 2008), which contains a large number of general knowledge facts. WN18 is a subset of WordNet, a KG featuring lexical relations between words. FB15k and WN18 are the most widely used datasets in the task of KGC. Recent studies (Toutanova and Chen 2015; Dettmers et al. 2018) found that the two datasets contain inverse relations. Under the circumstances, FB15k-237 and WN18RR were proposed, which removed the reverse relations in FB15k and WN18, and are regarded as more challenging datasets. The statistics of the datasets are summarized is Table 1.

Table 1: Statistics of the Four Datasets.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	#triples in Train/Valid/Test
FB15k	14,951	1,345	483,142 / 50,000 / 59,071
WN18	40,943	18	141,442 / 5,000 / 5,000
FB15k-237	14,541	237	272,115 / 17,535 / 20,466
WN18RR	40,943	11	86,835 / 3,034 / 3,134

In order to further analyze the performance of RGHAT on entities with different degrees. We split the entities into three sets according to their degrees in the training set. The first set $\mathcal{E}_1^{train}$ is made up of entities with top 10% degree values, the second set $\mathcal{E}_2^{train}$ consists of entities with top 10% to 50% degree values, and the third set $\mathcal{E}_3^{train}$ includes the remaining entities. Then, we obtain three sets from the test set in the following way. For each triple (h, r, t) in the test set, if $h \in \mathcal{E}_1^{train}$ and $t \in \mathcal{E}_1^{train}$, then $(h, r, t) \in \mathcal{T}_1^{test}$. The second and third set $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$ can be obtained in the same way with $\mathcal{E}_2^{train}$ and $\mathcal{E}_3^{train}$. We run our models on $\mathcal{T}_1^{test}$, $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$ to further test the performance. The number of triples in $\mathcal{T}_1^{test}$, $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$ are shown in Table 2.

Table 2: Statistics of $\mathcal{T}_1^{test}$, $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$.

Dataset	$\lvert\mathcal{T}_1^{test}\rvert$	$\lvert\mathcal{T}_2^{test}\rvert$	$\lvert\mathcal{T}_3^{test}\rvert$
FB15k	8299	8839	2110
WN18	333	706	206
FB15k-237	2566	2691	799
WN18RR	213	461	243

Baselines

To demonstrate the effectiveness of our model, we compare results with the following baselines.

- TransE (Bordes et al. 2013): one of the most widely used KGC models.
- DistMult (Yang et al. 2015): a popular tensor factorization based model which uses a bilinear score function to compute scores of knowledge triples.
- ComplEx (Trouillon et al. 2016): an extension of DistMult which embeds entities and relations into complex vectors instead of real-valued ones.
- RotatE (Sun et al. 2019): a state-of-the-art model which defines each relation as a rotation from the head entity to the tail entity in the complex vector space.
- ConvE (Dettmers et al. 2018): a popular convolutional network based model.
- ConvKB (Nguyen et al. 2018): another state-of-the-art convolutional network based model.
- R-GCN (Schlichtkrull et al. 2018): an extension of graph convolutional network, which can effectively model multi-relational data.
- A2N (Bansal et al. 2019): a recent model that learns query-dependent representations of entities based on a GNN structure.
- Nathani's (Nathani et al. 2019): a recent model that models the local neighborhood via graph attention network.

Experimental Settings

In the training stage, we adopt a two-layer RGHAT to train the entity and relation embeddings. For the encoder, the embedding size of entities is set as 100 for both the input and output layer. The number of heads for the multi-head attention mechanism is set as 8. And the bi-interaction combination is utilized by the information aggregator. In addition, a dropout with the rate as 0.5 is applied to each input layer of the encoder and the the normalized attention coefficients following graph attention network (Veličković et al. 2018). For the decoder, we also set the embedding size of entities and relations as 100 to be consistent with the encoder. A dropout with the rate as 0.3 is applied to the feature maps. During the training procedure, the learning rate is set as 0.0005 for FB15k and FB15k-237, and 0.0001 for WN18 and WN18RR. We apply L2 regularization with $\lambda = 0.0005$ for all the training parameters. And all the training parameters are randomly initialized.

In the test phase, we replace the head and tail entities with all the entities in KG in turn for each triple in the test set. Then we compute a score for each corrupted triple, and rank all the candidate entities according to the scores. Specifically, positive candidates are supposed to precede negative ones. Finally, the rank of the correct entity is stored. We compare our models with baselines using the following metrics: (1) Mean Rank (MR, the mean of all the predicted ranks); (2) Mean Reciprocal Rank (MRR, the mean of all the reciprocals of predicted ranks); (3) Hits@n (the proportion of ranks not larger than n). Lower values of MR and larger values of MRR and Hits@n indicate better performance. We report results in the "filtered" setting (Bordes et al. 2013).

Table 3: Experimental results on FB15k and WN18. ¶ indicates the results are taken from (Sun et al. 2019). The results of R-GCN are directly taken from the original paper.

	FB15k					**WN18**				
			Hits@N					Hits@N		
	MR	MRR	@1	@3	@10	MR	MRR	@1	@3	@10
TransE (Bordes et al. 2013)¶	-	0.463	0.297	0.578	0.749	-	0.495	0.113	0.888	0.943
DistMult (Yang et al. 2015)¶	42	0.798	-	-	0.893	665	0.797	-	-	0.946
ComplEx (Trouillon et al. 2016)¶	-	0.692	0.599	0.759	0.840	-	0.941	0.936	0.945	0.947
RotatE (Sun et al. 2019)¶	40	0.797	0.746	0.830	0.884	**309**	0.949	0.944	**0.952**	0.959
ConvE (Dettmers et al. 2018)¶	51	0.657	0.558	0.723	0.831	374	0.943	0.935	0.946	0.956
R-GCN (Schlichtkrull et al. 2018)	-	0.696	0.601	0.760	0.842	-	0.819	0.697	0.929	**0.964**
RGHAT (Ours)	**37**	**0.812**	**0.760**	**0.843**	**0.898**	342	**0.954**	**0.949**	0.951	**0.964**

Table 4: Experimental results on FB15k-237 and WN18RR. ¶ indicates the results are taken from (Sun et al. 2019). § indicates the results are taken from (Nathani et al. 2019). The results of A2N are directly taken from the original paper.

	FB15K-237					**WN18RR**				
			Hits@N					Hits@N		
	MR	MRR	@1	@3	@10	MR	MRR	@1	@3	@10
TransE (Bordes et al. 2013)¶	357	0.294	-	-	0.465	3384	0.226	-	-	0.501
DistMult (Yang et al. 2015)¶	254	0.241	0.155	0.263	0.419	5110	0.43	0.39	0.44	0.49
ComplEx (Trouillon et al. 2016)¶	339	0.247	0.158	0.275	0.428	5261	0.44	0.41	0.46	0.51
RotatE (Sun et al. 2019)¶	**177**	0.338	0.241	0.375	0.533	3340	0.476	**0.428**	0.492	0.571
ConvE (Dettmers et al. 2018)¶	244	0.325	0.237	0.356	0.501	4187	0.43	0.40	0.44	0.52
ConvKB (Nguyen et al. 2018)§	216	0.289	0.198	0.324	0.471	**1295**	0.265	0.058	0.445	0.558
R-GCN (Schlichtkrull et al. 2018)§	600	0.164	0.10	0.181	0.30	6700	0.123	0.08	0.137	0.207
Nathani's (Nathani et al. 2019)§	210	0.518	0.46	0.54	0.626	1940	0.44	0.361	0.483	0.581
A2N (Bansal et al. 2019)	-	0.317	0.232	0.348	0.486	-	0.45	0.42	0.46	0.51
RGHAT (Ours)	196	**0.522**	**0.462**	**0.546**	**0.631**	1896	**0.483**	0.425	**0.499**	**0.588**

Experimental Results

Experimental results are shown in Table 3 and Table 4. From Table 3 and Table 4, we have the following findings. (i) The results indicate that RGHAT significantly and consistently outperforms all the state-of-the-art competitors on four benchmark datasets. In both Table 3 and Table 4, RGHAT achieves the best results on most metrics. The experimental results clearly demonstrate the effectiveness of the proposed model. (ii) Specifically, comparing RGHAT with the decoder only model ConvE, we find that RGHAT achieves substantial improvements against ConvE on all the metrics, e.g., on FB15k-237, RGHAT outperforms ConvE with a margin as large as 0.206 on MRR. This result verifies the effectiveness of the encoder, and indicates the information learned from local neighborhood is valuable. It is worth noting that we also tried different models as the decoder of RGHAT including DistMult and ConvKB, and found using ConvE achieved the best performance. Specifically, when using DistMult as the decoder, the result of RGHAT is better than R-GCN, which also utilizes a DistMult Decoder; when using ConvKB as the decoder, RGHAT also significantly outperforms all the baselines, including Nathani's, which also uses ConvKB as the decoder. These results indicate the superiority of our encoder, and again validates the effectiveness of RGHAT.

In addition, in order to further analyze the effect of local neighborhood information on entities with different degrees, we compare the performance of RGHAT with the decoder only model ConvE, and the results are shown in Table 5 and Table 6. From these two tables, we have the following findings. (i) The proposed model clearly outperform ConvE across the board, which indicates entities with different degrees can all benefit from the local neighborhood information. (ii) We find that RGHAT achieves the greatest improvements against ConvE on triples with high-degree entities. Taking the metric of MRR as an example, RGHAT gets the improvements of 0.174, 0.151 and 0.134 on $\mathcal{T}_1^{test}$, $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$ of FB15k, respectively, compared to the decoder only model ConvE. The results indicate that high-degree entities can better benefit from the local neighborhood information. We conjecture the reason lies in that high-degree entities are surrounded by more neighboring triples, which provide more information than the neighborhood of low-degree ones.

Table 5: Experimental results on $\mathcal{T}_1^{test}$, $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$ of FB15k and WN18.

	FB15k						WN18					
	$\mathcal{T}_1^{test}$		$\mathcal{T}_2^{test}$		$\mathcal{T}_3^{test}$		$\mathcal{T}_1^{test}$		$\mathcal{T}_2^{test}$		$\mathcal{T}_3^{test}$	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
ConvE	0.669	0.842	0.643	0.826	0.628	0.807	0.949	0.959	0.941	0.956	0.933	0.942
RGHAT	**0.843**	**0.922**	**0.794**	**0.884**	**0.762**	**0.851**	**0.962**	**0.975**	**0.949**	**0.968**	**0.936**	**0.948**

Table 6: Experimental results on $\mathcal{T}_1^{test}$, $\mathcal{T}_2^{test}$ and $\mathcal{T}_3^{test}$ of FB15k-237 and WN18RR.

	FB15k-237						WN18RR					
	$\mathcal{T}_1^{test}$		$\mathcal{T}_2^{test}$		$\mathcal{T}_3^{test}$		$\mathcal{T}_1^{test}$		$\mathcal{T}_2^{test}$		$\mathcal{T}_3^{test}$	
	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10	MRR	Hits@10
ConvE	0.332	0.511	0.324	0.498	0.308	0.464	0.436	0.531	0.431	0.521	0.412	0.487
RGHAT	**0.539**	**0.654**	**0.522**	**0.626**	**0.488**	**0.574**	**0.492**	**0.602**	**0.477**	**0.579**	**0.451**	**0.534**

Table 7: Case study for entity *Los Angeles Lakers*.

Head Entity	Los Angeles Lakers
Relation	sports_team_location, $\alpha_{h,r} = 0.0475$
Tail Entity	Los Angeles, $\beta_{r,t} = 1.0$, $\mu_{h,r,t} = 0.0475$
Relation	sports_team_roster/player, $\alpha_{h,r} = 0.2153$
Tail Entities	Magic Johnson, $\beta_{r,t} = 0.3082$, $\mu_{h,r,t} = 0.0664$ Kobe Bryant, $\beta_{r,t} = 0.2864$, $\mu_{h,r,t} = 0.0617$ Lamar Odom, $\beta_{r,t} = 0.0564$, $\mu_{h,r,t} = 0.0121$ ...
Relation	sports_team_roster/position, $\alpha_{h,r} = 0.0098$
Tail Entities	power forward, $\beta_{r,t} = 0.2661$, $\mu_{h,r,t} = 0.0026$ shooting guard, $\beta_{r,t} = 0.2428$, $\mu_{h,r,t} = 0.0024$ ...

Table 8: Case study for entity *Tom Hanks*.

Head Entity	Tom Hanks
Relation	award_winner^{-1}, $\alpha_{h,r} = 0.1292$
Tail Entities	56th Golden Globe Awards, $\beta_{r,t} = 0.1261$, $\mu_{h,r,t} = 0.0163$ 59th Golden Globe Awards, $\beta_{r,t} = 0.1332$, $\mu_{h,r,t} = 0.0172$ 66th Golden Globe Awards, $\beta_{r,t} = 0.1196$, $\mu_{h,r,t} = 0.0155$ ...

Case Studies

We also provide some case studies for RGHAT. Table 7 shows some neighboring relations with corresponding tail entities for the head entity *Los Angeles Lakers*. $\alpha_{h,r}$, $\beta_{r,t}$ and $\mu_{h,r,t}$ represent the relation-level, entity-level and the triple-level attention scores, respectively. From Table 7, we find that the relation *sports_team_roster/player* plays a more important role than *sports_team_location*, and both the two relations are much more important than the relation *sports_team_roster/position*. Among the tail entities of *sports_team_roster/player*, we find top basketball players *Magic Johnson* and *Kobe Bryant* are the most weighted[1]. These observations are in line with our intuition, which indicates that the proposed model is capable to well distinguish the importance of local neighborhood. Moreover, the hierarchical attention mechanism increases the interpretability of our model, making it possible for us to tell how an entity aggregates information from the neighborhood.

Furthermore, we find an interesting phenomenon when comparing results of RGHAT with the recent neighborhood-aware model Nathani's (Nathani et al. 2019). It is observed that Nathani's tends to assign totally different weights to triples that are intuitively similar in importance, e.g., in the results of Nathani's, the weights of triple *(Los Angeles Lakers, sports_team_roster/player, Magic Johnson)* and *(Los Angeles Lakers, sports_team_roster/player, Kobe Bryant)* are 0.0759 and 0.0108, respectively. The former is almost 7 times of the latter, which is obviously inconsistent with our intuition. Our model assigns the weights of 0.0664 and 0.0617 to the two triples (see Table 7), which is considered to be more suitable and is consistent with our intuition. The same phenomenon are observed by the entity *Tom Hanks*. Our model RGHAT assigns similar weights for the 3 triples in Table 8. While Nathani's assigns 0.0253, 0.0177, 0.0026 for the three triples, and the weight of the first triple is almost 10 times of the third one. These observations indicate that compared to other state-of-the-art neighborhood-aware models, on account of our utilizing the hierarchical attention mechanism, the weights of neighboring triples with the same relations can be computed in a collective way, making our model more stable than state-of-the-art competitors and more consistent with human intuition.

Conclusion

In this paper, we proposed a novel neighborhood-aware model RGHAT for the KGC task. RGHAT is equipped with a hierarchical attention mechanism, which can effectively aggregate the local neighborhood information of each entity. Particularly, the hierarchical attention mechanism provides a fine-grained learning process for the proposed model, which increased the interpretability of RGHAT. Moreover, further analysis showed the results of RGHAT were more consis-

[1]In Freebase, the tail entities of *sports_team_roster/player* include all the players that have played for the team.

tent with human intuition compared to other neighborhood-aware models. Finally, extensive experiments on popular benchmarks clearly validated the superiority of RGHAT against various sate-of-the-art baselines.

Acknowledgements

The research work is supported by the National Key Research and Development Program of China under Grant No. 2017YFB1002104, the National Natural Science Foundation of China under Grant No. U1836206, U1811461, 61773361, 61836013 the Project of Youth Innovation Promotion Association CAS under Grant No. 2017146.

References

Ashburner, M.; Ball, C. A.; Blake, J. A.; Botstein, D.; Butler, H.; Cherry, J. M.; Davis, A. P.; Dolinski, K.; Dwight, S. S.; Eppig, J. T.; et al. 2000. Gene ontology: tool for the unification of biology. *Nature genetics* 25(1):25.

Bansal, T.; Juan, D.-C.; Ravi, S.; and McCallum, A. 2019. A2n: Attending to neighbors for knowledge graph inference. In *ACL*, 4387–4392.

Bollacker, K.; Evans, C.; Paritosh, P.; Sturge, T.; and Taylor, J. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 1247–1250.

Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; and Yakhnenko, O. 2013. Translating embeddings for modeling multi-relational data. In *NIPS*, 2787–2795.

Carlson, A.; Betteridge, J.; Kisiel, B.; Settles, B.; Hruschka, E. R.; and Mitchell, T. M. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

Dalton, J.; Dietz, L.; and Allan, J. 2014. Entity query feature expansion using knowledge base links. In *SIGIR*, 365–374.

Dettmers, T.; Minervini, P.; Stenetorp, P.; and Riedel, S. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI*.

Evgeniou, T., and Pontil, M. 2004. Regularized multi–task learning. In *KDD*, 109–117. ACM.

Ferrucci, D.; Brown, E.; Chu-Carroll, J.; Fan, J.; Gondek, D.; Kalyanpur, A. A.; Lally, A.; Murdock, J. W.; Nyberg, E.; Prager, J.; et al. 2010. Building watson: An overview of the deepqa project. *AI magazine* 31(3):59–79.

García-Durán, A.; Bordes, A.; and Usunier, N. 2015. Composing relationships with translations. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 286–290.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lin, Y.; Liu, Z.; Luan, H.; Sun, M.; Rao, S.; and Liu, S. 2015a. Modeling relation paths for representation learning of knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 705–714.

Lin, Y.; Liu, Z.; Sun, M.; Liu, Y.; and Zhu, X. 2015b. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, 2181–2187.

Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *ACL-IJCNLP*, 1003–1011. Association for Computational Linguistics.

Nathani, D.; Chauhan, J.; Sharma, C.; and Kaul, M. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs.

Nguyen, D. Q.; Nguyen, T. D.; Nguyen, D. Q.; and Phung, D. 2018. A novel embedding model for knowledge base completion based on convolutional neural network. In *NAACL-HLT*, 327–333.

Nickel, M.; Tresp, V.; and Kriegel, H.-P. 2011. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, 809–816.

Schlichtkrull, M.; Kipf, T. N.; Bloem, P.; Van Den Berg, R.; Titov, I.; and Welling, M. 2018. Modeling relational data with graph convolutional networks. In *ESWC*, 593–607. Springer.

Suchanek, F. M.; Kasneci, G.; and Weikum, G. 2007. Yago: a core of semantic knowledge. In *WWW*, 697–706.

Sun, Z.; Deng, Z.-H.; Nie, J.-Y.; and Tang, J. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *ICLR*.

Toutanova, K., and Chen, D. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, 57–66.

Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; and Bouchard, G. 2016. Complex embeddings for simple link prediction. In *ICML*, 2071–2080.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*, 5998–6008.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2018. Graph attention networks. In *ICLR*.

Wang, Z.; Zhang, J.; Feng, J.; and Chen, Z. 2014. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, 1112–1119.

Yang, B.; Yih, W.-t.; He, X.; Gao, J.; and Deng, L. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. 2016. Hierarchical attention networks for document classification. In *NAACL-HLT*, 1480–1489.

Yuan, M., and Lin, Y. 2006. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 68(1):49–67.