One-shot Relation Learning for Knowledge Graphs via Neighborhood Aggregation and Paths Encoding

JIAN SUN, School of Artificial Intelligence, University of Chinese Academy of Sciences and National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing 100049, P.R.China

YU ZHOU, National Laboratory of Pattern Recognition, Institute of Automation, CAS, School of Artificial Intelligence, University of Chinese Academy of Sciences, and Beijing Fanyu Technology Co., Ltd, Beijing 100049, P.R.China

CHENGQING ZONG, National Laboratory of Pattern Recognition, Institute of Automation, CAS and School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, P.R.China

The relation learning between two entities is an essential task in knowledge graph completion that receives much attention recently. Previous work most exclusively focused on relations widely seen in the original knowledge graphs (KGs), which means enough training data are available for modeling. However, long-tail relations that only show in a few triples are actually much more common in practical KGs. Without sufficiently large training data, the performance of existing models on predicting long-tail relations drops impressively. This work aims to predict the relation under a challenging setting where only one instance is available for training. We propose a path-based one-shot relation prediction framework, which can extract neighborhood information of an entity based on the relation query attention mechanism to learn transferable knowledge among the same relation. Simultaneously, to reduce the impact of long-tail entities on relation prediction, we selectively fuse path information between entity pairs as auxiliary information of relation features. Experiments in three one-shot relation learning datasets show that our proposed framework substantially outperforms existing models on one-shot link prediction and relation prediction.

CCS Concepts: • **Computing methodologies** → *Machine learning*; **Knowledge representation and reasoning**; *Natural language processing*.

Additional Key Words and Phrases: one-shot learning, relation learning, knowledge graph completion

ACM Reference Format:
JIAN SUN, YU ZHOU, and CHENGQING ZONG. 2021. One-shot Relation Learning for Knowledge Graphs via Neighborhood Aggregation and Paths Encoding. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* 1, 1 (September 2021), 20 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

In recent years, more and more large scale knowledge graphs [1, 3, 17, 18] have been constructed. They have been widely used in many fields, such as dialogue system [33], machine translation [42] and entity linking [36]. However, with the

Authors' addresses: JIAN SUN, School of Artificial Intelligence, University of Chinese Academy of Sciences and and National Laboratory of Pattern Recognition, Institute of Automation, CAS, Beijing 100049, P.R.China, jian.sun@nlpr.ia.ac.cn; YU ZHOU, National Laboratory of Pattern Recognition, Institute of Automation, CAS, and School of Artificial Intelligence, University of Chinese Academy of Sciences, and and Beijing Fanyu Technology Co., Ltd, Beijing 100049, P.R.China, yzhou@nlpr.ia.ac.cn; CHENGQING ZONG, National Laboratory of Pattern Recognition, Institute of Automation, CAS and and School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, P.R.China, cqzong@nlpr.ia.ac.cn.

rapid expansion and evolution of world knowledge, new entities and relations that are scarcely or even never covered by established knowledge bases keep emerging. This means a large portion of KG relations are actually long-tail. As shown in Figure 1, there is a large portion of relations that only have a few triples, and most deep neural-network-based methods are difficult to deal with such data. In this paper, we aim to predict long-tail relations. Especially, we will tackle the most challenging one-shot case, where only one sample is available for relation prediction.

The one-shot learning has been widely developed in KGs [4, 6, 14, 16, 24, 35, 40]. In knowledge graph completion, [35] utilized the knowledge extracted by embedding models and learned a matching metric by considering both the learned embedding and one-hop graph structures. [40] and [24] are two different few-shot relation learning frameworks. [40] designed a recurrent autoencoder aggregation network to model interactions of few-shot reference entity pairs and accumulate their expression capabilities for each relation. [24] adopted a stack of Transformer blocks to model interactions of reference entity pairs. [40] and [24] enhance the entity embedding module in [35] and design a static and dynamic attention aggregation module respectively. However, these above-mentioned methods focused on tail entity prediction. They didn't explore the one-shot relation predicting, which is the focus of this paper. Meanwhile, the above methods take the representation of entities on both sides of the relation as the basis for the representation of the relation, so it has relatively large requirements on the quality of entity embedding. For some newly added entities and entities with few instances in the knowledge base (long-tail entities), the above methods are difficult to obtain a good entity representation for these long-tail entities.

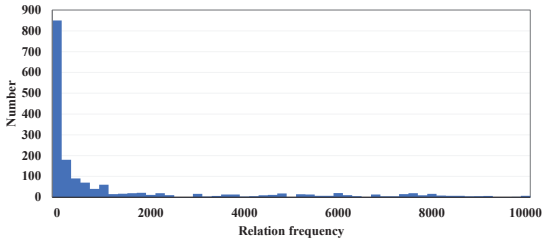


Fig. 1. The histogram of relation frequencies in Wikidata.

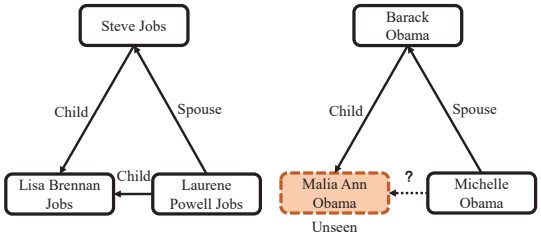


Fig. 2. An illustration that relation between an unseen entity pair can be inferred by the information of paths.

In KGs, the characteristics of a relation can be represented by the entities at both ends. At the same time, the relations can also reflect the characteristics of the entity to some extent. For example, if the relation connected to an entity is `Gender`, the entity may be two different human names, but it must not be a location name. According to it, we design a novel attention-based neighborhood aggregation method to encode entity, which uses the average of the relations as the query and the relations as the key to design the attention mechanism.

Meanwhile, we model paths between entities as auxiliary information for relation representation to deal with invisible entities. We identify all paths from the head entity to the tail entity in KGs. Each path is represented by its relation. The paths modeling is beneficial. In Figure 2, on the left is the training example, and on the right is the test example. The entity `Malia Ann Obama` is unseen during training; it is difficult to get a good representation of the entity `Malia Ann Obama`. Nevertheless, we can infer the missing relation directly from the hidden rules in the path. For instance, the embedding of a path (`Spouse, Child`) has a high similarity with relation `Child` after training. During the test, even if the entity `Malia Ann Obama` does not have a good representation, we can correctly infer `Malia Ann Obama` is `Michelle Obama`'s child by the path (`Spouse, Child`).

We combine neighborhood aggregation embedding and path embedding as the joint embedding feature of the relations. Then, we input the joint embedding into the one-shot relation prediction framework. We design two one-shot relation learning methods, prototypical network based method and convolution neural network based method. The difference between the two methods is that the one-shot relation prediction framework based on the prototype network has no parameters itself.

In summary, our main contributions as follows:

- We propose a one-shot relation prediction model that can predict the long-tail relation between entities.
- Different from other one-shot knowledge graph completion, we model paths between entities as auxiliary information for relation representation to deal with invisible entities.
- We design a new training and testing method for one-shot relation learning, which can predict relations and entities at the same time.
- Considered there are still few datasets specifically built for one-shot knowledge graph completion, we collect a new dataset based on FB15k [3] dataset.
- The proposed method is evaluated on three datasets for one-shot knowledge graph completion. We report state-of-the-art results under most of the experimental settings in these tasks.

2 RELATED WORK

2.1 Embedding Models

Various embedding-based models have been developed to complete knowledge graphs automatically. The models project the entities and relations into a low-dimensional vector space and define a score function to measure the plausibility of each triple. The total likelihood of triples is finally maximized to learn the embedding of entities and relations. TransE [3] regarded each relation as a translation vector between the head embedding and the tail embedding. Based on the idea of TransE, more advanced models such as TransH [34], RotatE [26] have been proposed. TransH [34] introduced the hyperplane with a normal vector to make the entities with different relations have different representation. RotatE [27] was able to model and infer various relation patterns including symmetry, antisymmetry, inversion, and composition. It defined each relation as a rotation from the source entity to the target entity in the complex vector space. There are other embedding-based models like RESCAL [20], DistMult [37], TuckER [2], and ComplEx [29], which optimized a scoring function with a bilinear product between vector embedding for each subject and object entity and a full rank matrix for each relation.

Recently, various path-based embeddings models have been explored, for instance, a random walk relation inference model, PRA [13], designed for predicting new relation instances in KGs. Path-RNN [19] uses a recurrent neural network (RNN) to generate paths embedding and Das et al. [5] improves the performance of Path-RNN by multi-path combination operator. Wang et al. [32] adaptively integrate the entity context and relational path through a learnable attention mechanism to reason missing relation.

However, the above embedding-based models usually assume enough training instances for all relations and entities and do not pay attention to those long-tail relations. These models do not perform well in predicting long-tail relations without sufficiently large annotated data, especially large-scale KGs.

2.2 One-Shot Learning

Deep learning usually does not fare well in few data cases. There are new classes that have never been seen before, and only a few annotated samples can be used for each class. One-shot learning should be an effective method to solve these few data cases. It could be roughly divided into three categories: model-based, optimization-based, and metric-based.

The model-based method aims to quickly update parameters on a few number of samples through the design of the model structure and directly establish the mapping function of input and predicted value. For example, Santoro et al. [23] proposed the method of memory enhancement to solve the few-shot Learning task based on Neural Turing Machine, which could learn strategies of storing expressions into memory and predicted them according to these expressions.

The optimization-based [7, 15, 22] method considers that ordinary gradient descent method is difficult to fit in the few-shot scenario. It completes the task of small sample classification by adjusting the optimization method. Ravi and Larochelle [22] used LSTM to express meta learner and used its state to express the updating of target classifier parameters, and it learned how to initialize and update the learner network on the new classification task.

The metric-based method [12, 25, 30, 39] is to model the distance distribution between samples, so that similar samples are close to each other and different samples are far away from each other. for example, Siamese Neural Networks [12] and Prototype Networks [25]. Siamese Network [12] constructed different paired samples by combination and judged whether they belonged to the same class by the distance of the samples and used the features extracted from the network for few-shot learning. Prototype Network [25] was based on the idea that there existed a prototype expression for each category, which is the mean value of the support set in embedding space. For a query sample, the Euclidean distances between the query sample and all the support samples are calculated. The label of the least distance support samples is regarded as the category of the query sample.

In information extraction Zong et al. [43], one-shot relation extraction [8–10, 21, 38, 41] aims to classify the relation between two given entities based on their related context, which focuses on the text. In knowledge graph completion, we consider completing an entity or relation based on the graph structure. Under the framework of one-shot learning, entity completion has been widely studied [4, 6, 16, 24, 35, 40]. Lv et al. [16] adopted meta-learning to learn effective meta parameters from high-frequency relations that could quickly adapt to few-shot relations. Motivated by the dual process theory in cognitive science, Du et al. [6] proposed CogKR, which consisted of summary and reasoning modules. The summary module probed the underlying relationship of a given instance, while the reasoning module constructed cognitive a graph to infer the correct answer. MetaR [4] solved the few-shot link prediction by focusing on transferring relation-specific meta information to make models learn the most critical knowledge and learn faster.

However, the one-shot relation completion is still barely explored. From this viewpoint, we design the one-shot relation prediction framework based on neighborhood aggregation and paths encoding.

3 OUR APPROACHES

Knowledge graphs consist of a set of entities $\mathcal{E}$ and a set of relations $\mathcal{R}$. Knowledge facts are stored as a collection of triples $\mathcal{G} = \{(h, r, t)\}$, where $h \in \mathcal{E}$ is the head entity, $t \in \mathcal{E}$ is the tail entity and $r \in \mathcal{R}$ is the relation that connects the two entities.

In the task of knowledge graph completion, relation prediction is to predict relation r between two existing entities: $(h, ?, t)$. So the one-shot relation prediction can be formalized as : given an entity pair (h_r, t_r), the correct relation is

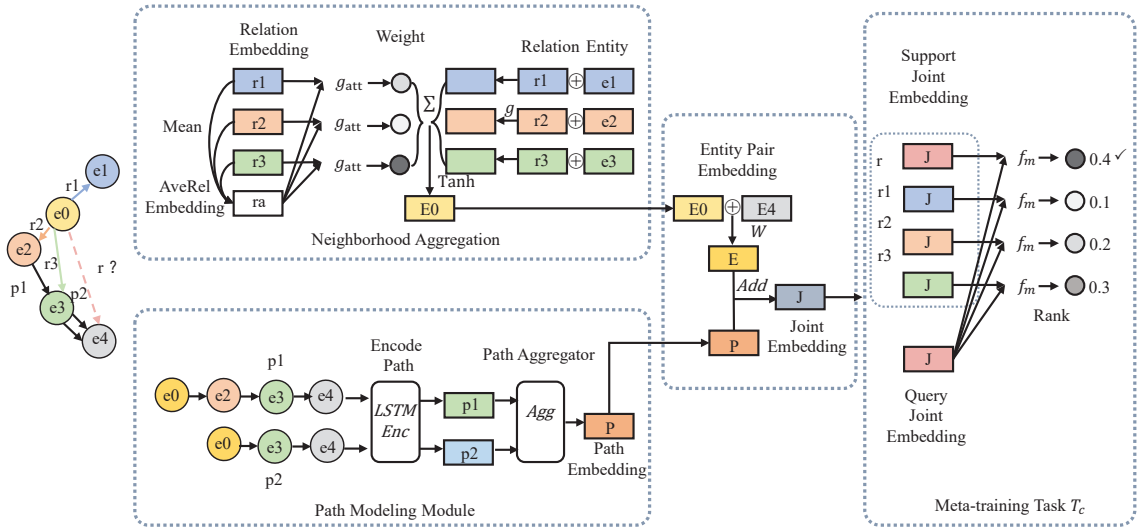


Fig. 3. An illustration of our one-shot relation prediction framework. AveRel-Embedding is the average embedding of adjacency relations. '⊕' denotes concatenation. 'f_m' is the metric function.

r. However, in KGs, there may be few or even one instance (h_0, r, t_0) about relation r. We need to utilize (h_0, r, t_0) to correctly predict the relation of (h_r, t_r).

The model proposed for one-shot relation prediction consists of three modules: the attention-based neighborhood aggregation module, the paths encoding module, and the one-shot relation prediction module. The attention-based neighborhood aggregation module is to encode entities in the knowledge graphs, and the path encoding module is to use the path as auxiliary information to alleviate the problem of insufficient entity encoding caused by long-tail entities. Our overall model is shown in Figure 3.

3.1 Attention-based Neighborhood Aggregation

This module is designed to enhance the representation of each entity by aggregating neighborhood information in knowledge graphs. In the local graph convolutional aggregation algorithms [35], the function of neighborhood aggregation is:

$$f_{\mathrm{GCN}}(\mathcal{N}_e) = \sigma\Big(\frac{1}{|\mathcal{N}_e|}\sum_{k=1}^{|\mathcal{N}_e|} g([\mathbf{r}_k; \mathbf{e}_k])\Big) \tag{1}$$

where $\mathbf{e}_k$ and $\mathbf{r}_k$ are the vector representation of r_k and e_k. $|\mathcal{N}_e|$ is number of relations, $\sigma(\cdot)$ is the activation function, $g(\cdot)$ is the full connection layer. We can see that all neighborhood sets of entities e are treated equally in representing e. However, most relationships between entity e and adjacent entities are different. Intuitively, they should contribute differently in representing e. So we build an attention mechanism in our model to select the relevant information in a designated graph neighborhood.

We design the attention-based neighborhood aggregation module based on the hypothesis that, in knowledge graphs, a certain amount of relation connected with an entity can reflect the characteristics of the entity to some extent. If the adjacency relation of one entity is similar to that of another entity, the characteristics of these two entities are also

relatively close. For example, entity `Steve Jobs` and entity `Elon Musk` may have similar adjacent relations, such as `Nationality`, `Gender`, `Company`, etc. However, the adjacency relation between entity `Steve Jobs` and entity `Car` may be quite different.

For any given entity e, the neighborhood set of e is as $\mathcal{N}_e = \{(r_k, e_k) \mid (e, r_k, e_k) \in \mathcal{G}\}$. The purpose of neighborhood aggregation is to encode $\mathcal{N}_e$ and output a vector as the latent representation of entity e. The function of neighborhood aggregation is:

$$f(\mathcal{N}_e) = \tanh(\sum_{k=1}^{|\mathcal{N}_e|} \alpha_k^e g([\mathbf{r}_k; \mathbf{e}_k])) \tag{2}$$

$$\alpha_k^e = \frac{\exp\left(g_{\mathbf{att}}\left([\mathbf{r}_k; q_e]\right)\right)}{\sum_{j=1}^{|\mathcal{N}_e|} \exp\left(g_{\mathbf{att}}\left([\mathbf{r}_j; q_e]\right)\right)} \tag{3}$$

$$q_e = \frac{1}{|\mathcal{N}_e|} \sum_{(\mathbf{r}_k) \in \mathcal{N}_e} \mathbf{r}_k \tag{4}$$

where $\mathbf{e}_k$ and $\mathbf{r}_k$ are the vector representation of r_k and e_k with dimension d. $|\mathcal{N}_e|$ is number of relations, $\tanh(\cdot)$ is the activation function, $g(\cdot)$ and $g_{\mathbf{att}}(\cdot)$ are the full connection layers.

We use the average of the relations as query and the relations as key, giving different weights to different relations. At the same time, when the final entity embedding is generated according to the weight, we also fuse the information of the adjacent entity. Therefore, the embedding of the final entity contains common characteristics and unique characteristics that are different from other entities. It is worth noting that we only encode the local graph of the entity and perform one-step propagation. This ensures that the module is applied quickly to large knowledge graphs.

3.2 Paths Encoding

This module is designed to model paths between entity pairs, which is auxiliary information for relation representation. Given a pair of entities (h_r, t_r), $P = \{p_1, p_2, ..., p_u\}$ is a set of paths between pairs of entities, where $p_l = \{r_1, r_2, ..., r_L\}$ is l-th path, where L is the length of p_i. Note that we do not use the identity of entities when modeling paths, in order to enhance the representation of some entity pairs that contain unseen entities during training.

3.2.1 Paths learning. We adopt LSTM to encode paths into vector representations.

$$\mathbf{h}_l^j, \mathbf{c}_l^j = \text{LSTM}(\mathbf{r}_l^j, [\mathbf{h}_l^{j-1}, \mathbf{c}^{j-1}]) \tag{5}$$

where $\mathbf{h}_l^{j-1}$ and $\mathbf{c}_l^{j-1}$ is the hidden state and cell state, respectively. We use the final output state $\mathbf{h}_l^L$ of the LSTM as the representation of l-th path. The advantage of RNN for path embedding is that it can capture the similarity among different relational paths based on the sequence of relations.

3.2.2 Attention-based paths aggregation. Given the embedding of each path, then we design the attention-based path combination module.

$$\mathbf{o}_i^L, \mathbf{m}_i^L = \text{LSTM}(\mathbf{h}_i^L, [\mathbf{o}_{i-1}^L, \mathbf{m}_{i-1}^L]) \tag{6}$$

$$a_i = \text{softmax}(\mathbf{W_1} \mathbf{o}_i^L) \tag{7}$$

$$\mathbf{h}_{(h_r,t_r)}^{out} = \sum_{i=1}^{U} a_i \mathbf{o}_i^L \tag{8}$$

where h_i^L is the embedding of i-th path between entity pair (h_r, t_r). $\mathbf{o}_{i-1}^L$ and $\mathbf{m}_{i-1}^L$ is the hidden state and cell state. Note that we take a fixed number of paths for each entity pairs, here, it is U. $\mathbf{W_1} \in R^{d\times 1}$ are parameters to be learned. $\mathbf{h}_{(h_r,t_r)}^{out}$ is the final path representation after attention aggregation. It is worth noting that there is no sequential relationship between paths. We use the LSTM here only to obtain the attention weight between similar paths. We also use other paths aggregation methods, for example, the Mean-based paths aggregation method, the Max-Pooling-based paths aggregation method, and the Sum-based paths aggregation method.

3.2.3 Joint embedding. Finally, we combine the representation of entity pairs with the representation of paths as the characteristics of a relation.

$$\mathbf{RC}(h_r, t_r) = \mathbf{W_2}([f\left(\mathcal{N}_{h_r}\right); f\left(\mathcal{N}_{t_r}\right)]) + \mathbf{h}_{(h_r,t_r)}^{out} \tag{9}$$

where $\mathbf{W_2} \in R^{2d\times d}$ are parameters to be learned. Here we use a simple addition operation to merge the two parts of the information.

3.3 One-shot Relation Prediction

Given the attention-based neighborhood aggregation module and paths encoding module, now we discuss the one-shot relation prediction module.

The settings for one-shot learning are described in Algorithm 1. In order to simulate the one-shot learning mode of testing set, which predicts the relations of the entity pairs according to one relation instance, in the train stage, we sample randomly C relations from the collection of relation in training data. For each relation r_i in C, we select randomly one triple as the support triple $S_i = \{(s_{h_i}, r_i, s_{t_i})|i = 1, .., C\}$ and n triples as query triples $Q_{i,j} = \{(q_{h_j}, r_i, q_{t_j})|i = 1, ..., C, j = 1, ..., n\}$. Then we get a training task $T_c = \{S_i, Q_{i,j}\}$. All tasks in training from the meta-training set, denoted as $T_{mtr} = \{T_c\}$. Given a query entity pair $\mathbf{q}_k = (h_k, t_k)$ in $Q_{i,j}$, the one-shot learning method thus could be trained on the task set by ranking the support entity pair $\mathbf{s}_i = (h_i, t_i)$ in S_i. After training with meta-training set, the model can be used to predict relation in testing. The method of meta-testing is similar to meta-training.

In addition, we also suppose that the model has access to a background knowledge graph $\mathcal{G}'$ (see 4.1), which is a subset of $\mathcal{G}$. The relations among background knowledge graph, training set, validation set, and test set are disjoint, and the entities in the training set, validation set, and test set can be found in the background knowledge graph. The function of background knowledge graphs $\mathcal{G}'$ is to get the neighborhood set of entity e and generate paths between entity pairs. Figure 4 shows an example of the distribution of entities and relations in the dataset. Meanwhile, to enrich path information and neighborhood information, we add inverse triples to $\mathcal{G}'$, which have been proved to be effective in knowledge graphs completion.

We apply the function of neighborhood aggregation and paths encoding to the query entity pair $\mathbf{q}_k = (q_{h_k}, q_{t_k})$ in $Q_{i,j}$ and each support entity pair $\mathbf{s}_i = (h_i, t_i)$ in S_i, we get two joint vectors: $\mathbf{RC}(h_k, t_k)$ and $\mathbf{RC}(h_i, t_i)$. Next, we need to calculate the similarity between the two joint vectors, which is used to predict the relation.

In this section, we design the similarity metric function based on the fixed metric method and the learned metric method, respectively. In the fixed metric method, Euclidean distance is used as a similarity metric function. In the learned method, we design the metric function based on convolutional neural networks.

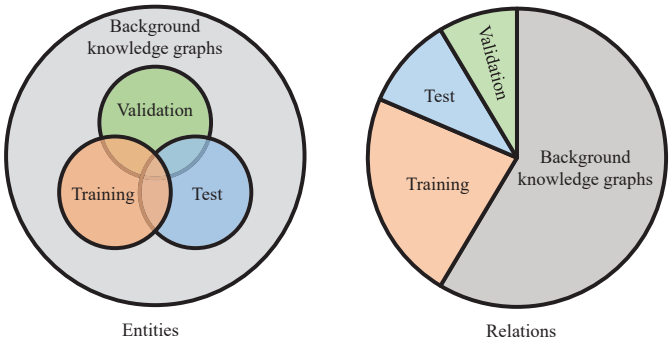


Fig. 4. An illustration of our dataset construction.

Algorithm 1 One-shot relation prediction

Input:
Background knowledge graphs Training data;
Initial parameter;

for epoch = 0:M **do**
  Sample C relations from the training data
  Empty support set S and query set Q
  for c in C **do**
    Sample one triple about c and add it to S
    Sample n triples about c and add to Q
  end for
  Get a training task $T = \{S, Q\}$
  for q_k in Q **do**
    for s_i in S **do**
      Calculate the metric scores based on joint embedding for query triple q_k and support triple s_i
    end for
  end for
  Calculate the loss
  Update parameter using gradient
end for

3.3.1 Fixed Metric Method. The relation prediction model based on the fixed metric used here is the prototypical networks [25]. The prototypical networks project the samples in each category into space. Then the mean of samples in this space is used as the prototypical center of this category.

In one-shot learning, there is only one sample data in each category, so the embedding of the instance is considered the prototypical center of this category.

$$\mathbf{c}_i = g_{\mathbf{enc}}\left(\mathbf{RC}(h_i, t_i)\right) \tag{10}$$

$$g_{\mathbf{enc}}(\mathbf{x}) = \mathbf{x} + Drop(\mathbf{W_3}(ReLU(\mathbf{W_4}\mathbf{x} + \mathbf{b_4})) + \mathbf{b_3}) \tag{11}$$

where $\mathbf{c}_i$ is the prototype for relation r_i, and $\mathbf{RC}(h_i, t_i)$ is the joint embedding of (h_i, t_i) in the triple $S_i = (h_i, r_i, t_i)$. $g_{\mathbf{enc}}(\cdot)$ is an encoder network, where $\mathbf{W_3}$ and $\mathbf{W_4}$ are learnable parameter, $\mathbf{b_3}$ and $\mathbf{b_4}$ are bias.

Then we calculate the Euclidean distance between the query entity pair and the prototypical center $\mathbf{c}_i$. We use Euclidean distance as a metric function here.

$$f_m(\mathbf{q}_k, \mathbf{c}_i) = ||g_{\mathbf{enc}}\left(\mathbf{RC}(h_k, t_k)\right) - \mathbf{c}_i||_2 \tag{12}$$

where $f_m(\mathbf{q}_k, \mathbf{c}_i)$ is the distance between the query entity pair $\mathbf{q}_k$ and the prototypical center $\mathbf{c}_i$. We perform relation prediction by ranking the distance, which can be considered as a multi-classification process. We use the cross-entropy loss function commonly used in multi-classification problems as the loss function of relation prediction. Remarkably, the distance is negatively correlated with similarity. The loss function is as follows,

$$loss = -\sum_{k=1}^{nC}\sum_{i=1}^{C} y_i \log \sigma(-f_m(\mathbf{q}_k, \mathbf{c}_i)) \tag{13}$$

where y_i is the label vector with dimension $R^{1\times 1}$, we apply the softmax function $\sigma = (\cdot)$ to the distance. Optimizing this loss function can help the approximate distance between query entity pair and support entity pair of the same relation.

The Euclidean metric method is simple and effective. However, the Euclidean metric is a fixed metric method. It only simply calculates the distance between two points in space and lacks the learning process.

3.3.2 Learned Metric Method. In this section, we design the learned metric method for one-shot relation prediction based convolutional neural networks (CNNs). Inspired from the success in computer vision [28], we use neural networks instead of fixed metric function. Following the one-shot learning settings, we first extract randomly C relation from the datasets and construct the support set $S = \{(h_i, r_i, t_i)\}_i^m$ by sampling randomly one entity pair for each relation. We extract a batch of entity pairs from the remaining data of the C relation as query sets $Q = \{(h_j, r_j, t_j)\}_i^n$. We first compute the representation of the support entity pair and the query entity pair based on neighborhood information of entity pairs:

$$\mathbf{s}_i = g_{\mathbf{enc}}(\mathbf{RC}(h_i, t_i)) \tag{14}$$

$$\mathbf{q}_k = g_{\mathbf{enc}}(\mathbf{RC}(h_k, t_k)) \tag{15}$$

Given the representations of the support entity pair and the query entity pair, next, we need to calculate the similarity between the two representations. We first combine the representations into 2D reshaping. It is treated as an input for a 2D convolutional layer. Using 2D convolutions can increase the expressiveness of the model through interaction between entities. The output of the convolutional layer is a vector. The vector is projected to a scalar by the full connection layers.

$$f_m(\mathbf{s}_i; \mathbf{q}_k) = f_\theta(f_\mathbf{c}([\mathbf{s}_i; \mathbf{q}_k])) \tag{16}$$

where the shape of $[\mathbf{s}_i; \mathbf{q}_k]$ is 2-D, $f_\mathbf{c}(\cdot)$ is the convolution function, $f_\theta(\cdot)$ are the full connection layers. Unlike the scoring function in fixed metric method, here the score is positively correlated with similarity.

Following the fixed metric method, we use the cross-entropy loss function as the loss function of relation prediction. The loss function is as follows

$$loss = -\sum_{k=1}^{nC}\sum_{i=1}^{C} y_i \log \sigma(f_m(\mathbf{s}_i; \mathbf{q}_k)) \tag{17}$$

where y_i is the label vector with dimension $R^{1\times 1}$, $\sigma = \text{softmax}(\cdot)$.

In the learned metric method, we capture the interaction between two entities through convolutional neural networks. Utilizing the interaction, the model can reasonably predict the deep relations between entities.

Dataset	Ent	Rel	Triples	Tasks
NELL-One	68,545	358	181,109	67
Wiki-One	4,838,244	822	5,859,240	183
FB15k-One	14,950	1212	508,791	68

Table 1. Statistics of datasets used in experiments. 'Tasks' denotes the number of relations we use as one-shot tasks. After removing the relations in 'Tasks', the facts of remaining relations are used as background knowledge graphs.

4 EXPERIMENTS AND RESULTS ANALYSIS

4.1 Datasets

We evaluate one-shot relation prediction on the NELL-One, Wiki-One, and FB15k-One, where the NELL-One and Wiki-One are created by Xiong et al. [35] based on real-world KGs NELL [18] and Wiki [31].

We construct FB15k-One dataset based on FB15k [3] dataset. We first count the number of triples for each relation in FB15k, then we select some relations as our task. In this paper, our task consists of 68 relations. We remove all triples associated with the 68 relations from the original FB15k to obtain the background knowledge graphs. We also need to make sure that all entities in the removed triples are visible in the background knowledge graphs. Finally, we divide the 68 relations and related triples into 53/5/10 to get the training set, the validation set and the test set.

The statistics of datasets are shown in Table 1. For NELL-One, we use 51/5/11 task relations for training/validation/testing. For Wiki-One, the division ratio is 133/16/34, and FB15k-One, the ratio is 53/5/10. Note that the Wiki-One dataset is an order of magnitude larger than any other benchmark datasets regarding the numbers of entities and triples.

Figure 5 is the statistics of paths between entities in the test set. In NELL-One, we can see that most pairs of entities in the test set do not have paths whose length is less than or equal to 2. As the length of the path increases, the path information between the entity pairs becomes more and more abundant. In Wiki-One, almost all entities have no path information between them, which means that most entities in the Wiki-One test data occur only once. In FB15k-One, path information between entities is significantly richer than that of NELL-One and Wiki-One.

4.2 Baseline Methods

In our experiments, we compare the following embedding-based methods: TransE [3], DistMult [37], ComplEx [29], RotatE [26]. For these models, we use the code released by Sun et al. [26]. We also experiment on the GMatching model [35]. As for CogKR and Meta-KGR models, they are specially designed to predict entity, so it is difficult to apply the model to relation prediction tasks. On the neighborhood aggregation module, we conducted some experiments to verify the effectiveness of attention-based neighborhood aggregation. Here, the neighbor aggregation baseline module contains the graph convolution neural encoding module (GCN) in Xiong et al. [35], the graph neural network encoding module (GNN) in Du et al. [6], the adaptive neighbor encoding module in FAAN [24], and the heterogeneous neighbors encoding in FSRL [40].

4.3 Some Details

We use the Adam [11] algorithm for training the model. Our models are implemented by PyTorch and run on NVIDIA GeForce GTX TITAN X Graphics Processing Units. We set the number of neighborhoods to 50 and the learning rate to 0.005. The epoch in experiments is 300. In the learning metric method, number of kernels is [16, 32] and kernel size is $[2 \times 2, 1 \times 2]$. The dimension of entity and relation embedding is set to 128 in NELL-One and FB15k-One. In Wiki-One,

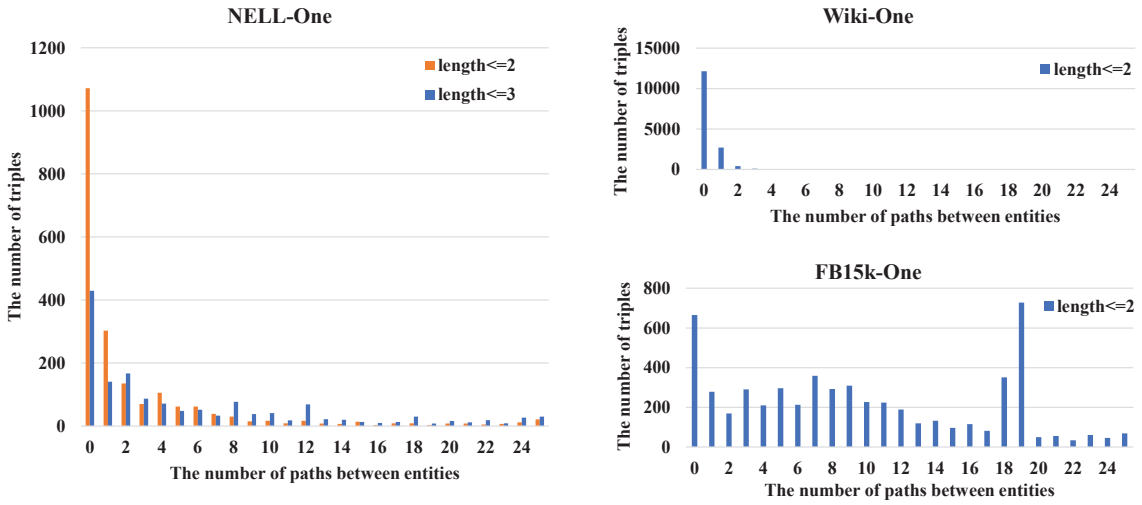


Fig. 5. Statistics of paths between entities in test set.

the dimension of embedding is 80. For paths encoding, the length of paths L is 2, and the number of paths U is 4. In training stage, the configuration of our models for the relation prediction task is $C = 20$, $n = 50$.

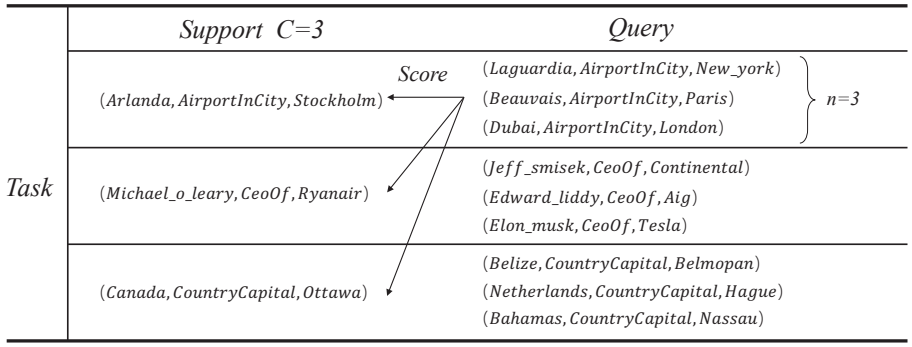
Figure 6 shows our relation prediction framework. We predict the relation between query entity pairs by sorting the triples in the support set. During training, we choose triples in $D_{r_{train}}$ as support triples without considering the triples in background knowledge graphs. However, During testing, the size C_{test} of the support set in meta-testing is equal to the number of all relations in the test set, training set and $\mathcal{G}'$. In other words, on FB15k-One, NELL-One, and Wiki-One, the size of the support set during testing is $1212 - 5 = 1207$, $358 - 5 = 353$, $822 - 16 = 806$, respectively.

In embedding-based methods, during training, we extract a triple randomly for each relation from testing set and add it to the dataset made up of background knowledge graph $\mathcal{G}'$ and training set, which together serve as the training set for the embedding models. The details can be seen Figure 7.

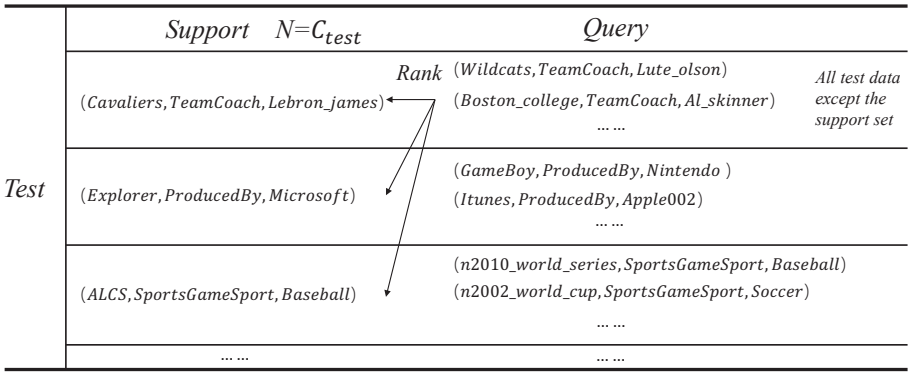
Figure 8 is an example of the baseline models GMatching [35], FRSL [40], FAAN [24]) for relation prediction task, where we change the test method to adapt to our tasks. These models construct a set of negative queries by randomly corrupting the tail entity of a positive triple. Then they use a hinge loss function to optimize their model. Since these models are used to predict entities, we design new test methods to fit our relation prediction task. The details can be seen Figure 8(b).

4.4 Results and Analysis

The main results of our module are shown in Table 2, KGs embedding baselines and GMatching one-shot learning results are shown at the top of the table. The results on fixed metric methods (FMetric) combined with different neighborhood aggregation methods are shown in the middle. We also show the results of combining the learned metrics method (LMetric) with different entity coding methods at the bottom of Table 2. The bold numbers denote the best results in the test. We use to mean reciprocal rank (MRR) and Hits@1 to evaluate different models, where MRR is the average of the reciprocal ranks of results and Hits@1 is the proportion of correct alignment ranked in top-1. Higher Hits@1 and MRR indicate better prediction performance.



(a) Training



(b) Testing

Fig. 6. Example of our relation prediction framework

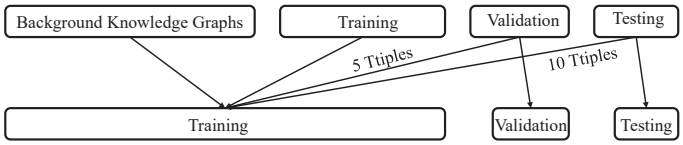
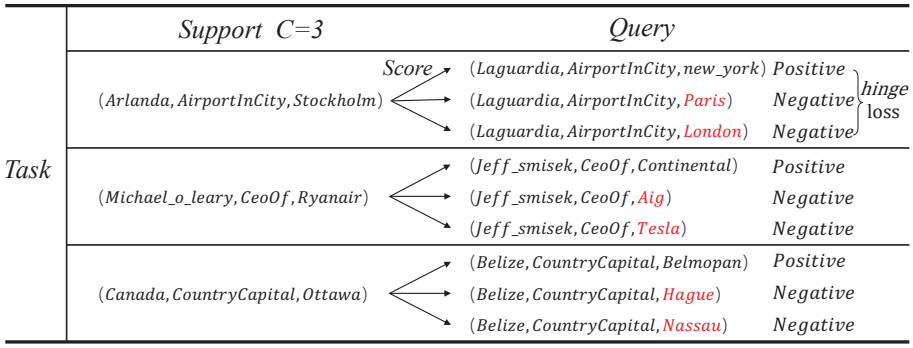
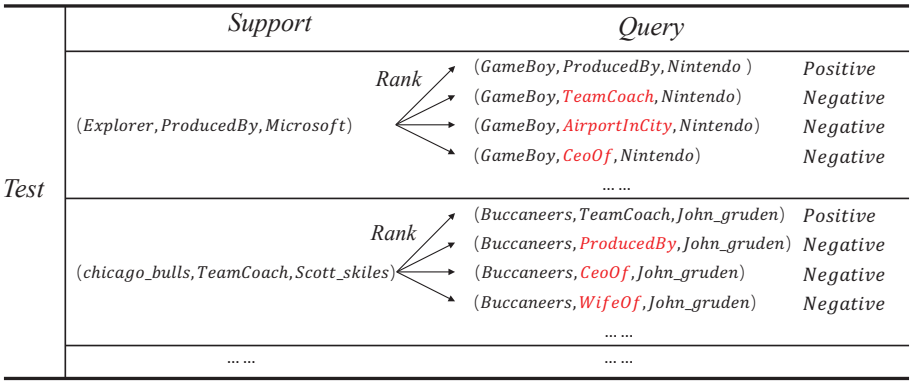


Fig. 7. Examples of data construction based on embedding-based methods for FB15k-One

We can see that the performance of the embedding models ComplEx on the small dataset is higher than that of GMatching. In FB15k-One, there is a lack of information about entity types, so GMatching doesn't apply here. However, on the large Wiki-One dataset, all the embedding models have poor performance. On the contrary, the improvement of GMatching is obvious on Wiki-One. It suggests that One-shot learning has more advantages than the embedded model in large datasets. Meanwhile, comparing the performance of FMetric and LMetric with GMatching, it can be seen that our relation prediction module has surprising results. In particular, in NELL-One, comparing GMatching, our GNN-FMetric model improves MRR value by a margin of 89%, Hits@1 value by a margin of 118%. The GCN-FMetric model improves MRR by a margin of 146%, and Hit@1 by margin 199%. The GNN-LMetric model improves MRR values



(a) Training



(b) Testing

Fig. 8. Example of baseline models for relation prediction

by a margin of 79% and increases Hit@1 values by 81%. The GCN-LMetric model improves MRR by a margin of 108%, and Hit@1 by margin 145%. In Wiki-One, comparing GMatching, the GNN-FMetric model improves MRR values by a margin of 41% and improves Hit@1 values by a margin of 33%. The GCN-FMetric model improves MRR by a margin of 87%, and Hit@1 by margin 92%. The GNN-LMetric model improves MRR values by a margin of 45% and increases Hit@1 values by 46%. The GCN-LMetric model improves MRR by a margin of 89%, and Hit@1 by margin 109%. It can also be seen from the above results that in relation prediction task, the training and testing mode of our model is better than that of GMatching. We can analyze the reasons according Figure 6 and Figure 8. In Figure 6, our training and testing processes are consistent. Since the relation between the triples in the support set is different, the model can distinguish the characteristics of different relations after training. However, in Figure 8, the meta-training task in GMatching don't take into account the differences between relations, which we think is the main reason for the poor performance.

4.4.1 One-shot learning methods. Next, focus on comparing the FMetric model with LMetric, as can be seen from Table 2, in NELL-One, the performance of the FMetric-based model is better than that of all LMetric-based model. In FB15k-One, MRR values and Hit@1 values have increased slightly. In Wiki-One, MRR values and Hit@1 values went up 2.6% and 8.4%, respectively. Under the same neighborhood aggregation method, the performance of the fixed metric

model	NELL-One		FB15k-One		Wiki-One	
	MRR	Hits@1	MRR	Hits@1	MRR	Hits@1
DistMult [37]	.392	.227	.187	.085	.008	.001
TransE [3]	.057	.012	.148	.101	.010	.002
ComplEx [29]	.468	.302	.212	.137	.009	.001
RotatE [26]	.066	.025	.078	.054	.009	.001
GMatching [35]	.262	.168	-	-	.179	.123
GNN-FMetric [6]	.494	.367	.369	.243	.252	.164
GCN-FMetric [35]	.645	.502	.559	.391	.335	.236
FSRL-FMetric [40]	.505	.365	.408	.306	.131	.089
FAAN-FMetric [24]	.522	.398	.334	.244	.121	.084
Att-FMetric	**.662**	**.523**	**.572**	**.394**	**.369**	**.263**
GNN-LMetric [6]	.469	.304	.465	.318	.259	.180
GCN-LMetric [35]	.544	.411	.582	.422	.338	.257
FSRL-LMetric [40]	.373	.221	.242	.129	.095	.048
FAAN-LMetric [24]	.353	.230	.197	.113	.089	.054
Att-LMetric	**.582**	**.431**	**.590**	**.424**	**.358**	**.267**

Table 2. Relation prediction results in different baseline models.

method is slightly superior to that of the learned metric method. It is worth noting that, once trained, our framework can be used to predict any newly added relations without fine-tuning.

4.4.2 Neighborhood aggregation methods. In neighborhood aggregation experiments based on FMetric, Att-FMetric outperforms all baseline aggregation methods on all datasets. In LMetric, we can get the same conclusion. We analyze that although the neighborhood relations are considered in updating entity embedding, only a simple average operation is performed, which results in all neighborhood relations some decisive relations being as important as some unimportant ones, thus causing some noise. Meanwhile, GNN-FMetric requires all entities to be present when training the model. Surprisingly, in the attention-based FSRL and FAAN, our relation prediction model do not perform well. We analyze that the main reason is that these models do not integrate relations into the final representation of entities, which may help in predicting the relations between entities. In general, attention aggregation is slightly better than other aggregation methods in both one-shot learning models.

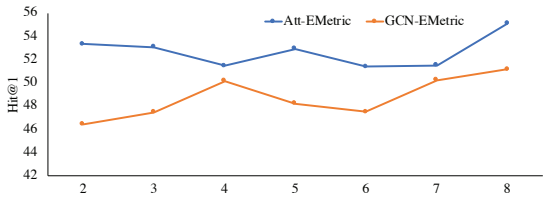


Fig. 9. Results of different number of paths on NELL-One.

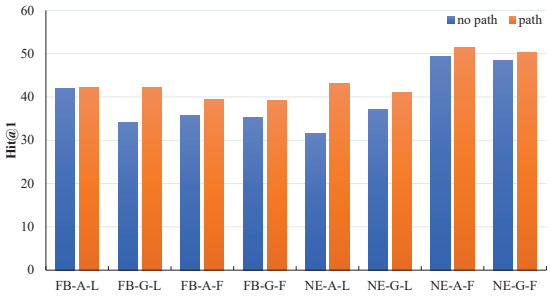


Fig. 10. Ablation studies on paths encoding.

Methods		FB15k-One		NELL-One		Wiki-One		Avg
		Hits@1	MRR	Hits@1	MRR	Hits@1	MRR	
Att-LMetric	max	0.336	0.5	0.332	0.476	0.185	0.302	0.355
	mean	0.317	0.486	0.368	0.518	0.219	0.323	0.372
	sum	0.317	0.474	0.344	0.498	0.189	0.301	0.354
	attn	**0.424**	**0.59**	0.431	0.582	0.267	0.358	0.442
GCN-LMetric	max	0.352	0.546	0.385	0.543	0.268	0.35	0.403
	mean	0.358	0.525	0.373	0.513	0.271	0.351	0.396
	sum	0.383	0.556	0.353	0.516	**0.271**	**0.353**	0.405
	attn	0.422	0.582	0.411	0.544	0.257	0.338	0.426
Att-FMetric	max	0.305	0.437	0.524	0.659	0.243	0.34	0.418
	mean	0.335	0.492	0.523	0.65	0.23	0.323	0.426
	sum	0.325	0.468	**0.524**	**0.666**	0.235	0.33	0.425
	attn	0.394	0.572	0.523	0.662	0.263	0.369	**0.464**
GCN-Fmetric	max	0.352	0.523	0.513	0.641	0.25	0.352	0.439
	mean	0.341	0.505	0.473	0.613	0.259	0.352	0.424
	sum	0.375	0.537	0.481	0.622	0.247	0.343	0.434
	attn	0.391	0.559	0.502	0.645	0.236	0.335	0.445

Table 3. Relation prediction results in different paths aggregation models.

4.4.3 Paths aggregation methods. We compare the four different paths aggregation methods, Attention-based paths aggregation, Mean-based paths aggregation methods, Max-Pooling-based paths aggregation method, and Sum-based paths aggregation methods. In FB15k-One, the best performance model is the model of combination Attention-based paths aggregation and Att-LMetric. In NELL-One, the Sum-based paths aggregation method is superior to other aggregation methods. In Wiki-One, we can get a conclusion similar to that in NELL-One. On the whole, the Attention-based paths aggregation method is better than the other three paths aggregation methods.

4.5 Ablation Study on Paths Encoding

4.5.1 Number of paths. We investigate the sensitivity of our model to the number of paths for each entity pair. The results on NELL-One are shown in Figure 9. We can see that increasing the number of paths does not significantly affect the results. This may be because a large number of entity pairs have little or no path with a length of 2 between them. For example, in the path set of the NELL-One database with a length of 2, '1702' entity pairs have no path, accounting for about 79%. This may cause that the number of paths is not very sensitive to the experimental results.

4.5.2 Ablation study on attention aggregation . In order to analyze the role of each component in attention aggregation, we conducted ablation experiments on FB15k-One. The results are shown in Table 4.

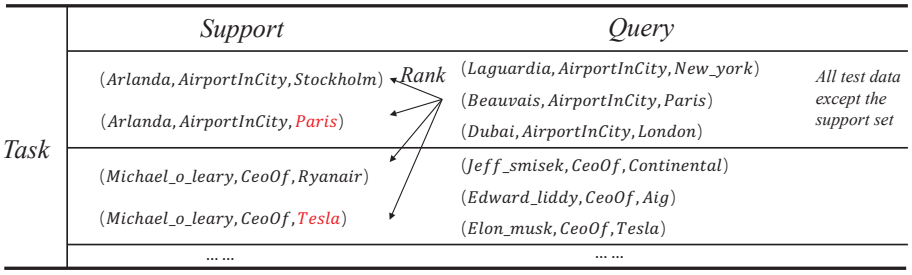
'w/o Average Relation' means that we use entities representation instead of average relation as the query. 'w/o Relation as Key' signifies that we use the entity embedding as key in equation 3. 'w/o Aggregate Relation' indicates that only tail entities are aggregated in the final information fusion. 'w/o Neighbor Encoder' represents the final representation of an entity without aggregated neighborhood information. We can see that both the averaged relations as query and relation as key play important roles in our model. Another important observation is that the aggregation of relation turns out to be essential.

Configuration	Hits@1
Att-LMetric	42.36%
w/o Relation as Key	41.08%
w/o Average Relation	35.48%
w/o Aggregate Relation	24.55%
w/o Neighbor Encoder	20.64%

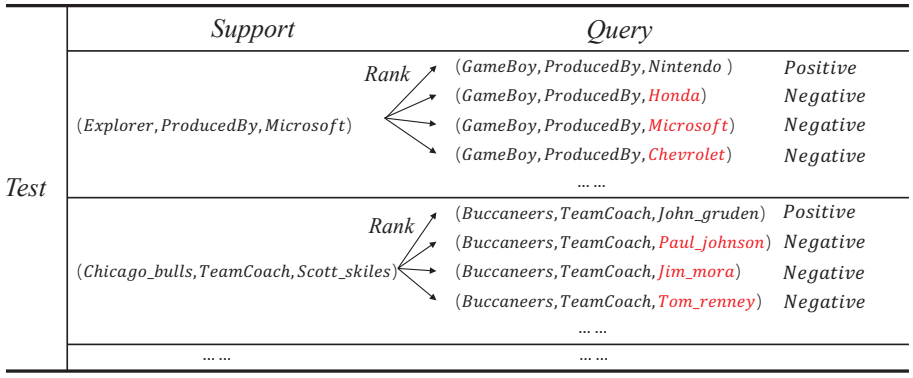
Table 4. Ablation studies on Att-LMetric.

We also construct the paths encoding ablation experiments. The results are shown in Figure 10, where 'FB' means FB15K-One, 'NE' is NELL-One, 'A' means attention, 'G' means GCN, 'L' is LMetric, and 'F' is FMetric. It is clear to see that path information can significantly increase the value of Hits@1. It shows that it is necessary to introduce path encoding as the supplementary information of relational characteristics.

4.6 Entity Prediction



(a) Training



(b) Testing

Fig. 11. Example of linking prediction test framework

We also apply our relation prediction framework to entity prediction. Figure 11 is our entity prediction training framework. We add a batch of negative instance triples for each triplet in the support set by corroding entities to form a

new support set. The results are shown in Table 5, where 'Best' is the best result of the different initializations , 'Rand' is the result of random initialization and all the results do not contain paths information.

model	Hits@1	Hits@5	Hit@10	MRR
RESCAL	.089	.186	.229	.140
TransE	.043	.141	.192	.093
DisMult	.066	.126	.177	.102
ComplEx	.086	.086	.223	.131
MetaR(Pre-train)	.093	.238	.331	.164
GMatching (Best)	.133	.260	.313	.188
GMatching (Rand)	.103	.186	.252	.151
GCN-FMetric (Rand)	.121	.230	.313	.177
Att-FMetric (Rand)	**.146**	**.293**	**.357**	**.213**

Table 5. One-shot for linking prediction on NELL-One.

The above results show that our relation prediction framework also has an excellent performance in the linking prediction tasks.

We also observe that the results on different relations are actually of high variance. For some relations such as `athleteInjuredHisBodypart`, `teamCoach` and `geopoliticalLocationOfPerson`, entities associated with these relations appear less frequently in KGs. These entities can be thought of as long-tail entities. Therefore, the existence of long-tail entities brings some difficulties to the linking prediction. Especially `teamCoach`, the tail entity is the name of the person, which is difficult to distinguish.

		MRR		Hits@10	
Relations	Candidates	GMatching	Att-FMetric	GMatching	Att-FMetric
sportsGameSport	123	.424	**.975**	**1.0**	.973
athleteInjuredHisBodypart	123	.025	**.031**	.015	**.088**
animalSuchAsInvertebrate	786	**.447**	.359	**.626**	.585
automobilemakerDealersInCountry	1084	**.453**	.318	**.821**	.789
sportSchoolIncountry	2100	**.534**	.484	.745	**.745**
politicianEndorsesPolitician	2160	.093	**.146**	.226	**.268**
agriculturalProductFromCountry	2222	.120	**.121**	.288	**.338**
producedBy	3174	.085	**.311**	.179	**.566**
automobilemakerDealersInCity	5716	.026	**.318**	.040	**.789**
teamCoach	10569	**.017**	.011	**.024**	.021
geopoliticalLocationOfPerson	11618	**.028**	.002	**.035**	.007

Table 6. Results decomposed over different relations.

Meanwhile, by comparing Table 7 and Figure 10 (NE-A-F and NE-G-F), we can see that the addition of negative instances does not affect the performance of relation prediction. Therefore, our model can perform relation prediction and linking prediction well at the same time.

model	Hits@1	MRR
GCN-FMetric (Negative sample)	.476	.606
Att-FMetric (Negative sample)	**.497**	**.644**

Table 7. One-shot for relation prediction on NELL-One.

5 CONCLUSION AND FUTURE WORK

This paper introduces a one-shot relation prediction framework based on attention neighborhood aggregation and paths encoding that could be used to predict long-tail relations in KGs. We design the attention mechanism based on an average relation query, so the embedding of the entity contains not only common features but also unique features. Meanwhile, we selectively fuse path information between entity pairs as auxiliary information to reduce the impact of long-tail entities. Then we design the one-shot relation prediction framework based on the fixed metric method and learned metric method. The experimental results on three datasets demonstrate that our one-shot relation learning framework is superior to the current embedding models and one-shot learning baseline model in link prediction and relation prediction. In future work, we will extend our model for integrating more abundant information.

REFERENCES

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *The Semantic Web*. Springer Berlin Heidelberg, Berlin, Heidelberg, 722–735.

[2] Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019. TuckER: Tensor Factorization for Knowledge Graph Completion. *arXiv preprint arXiv:1901.09590* (2019).

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., 2787–2795. http://papers.nips.cc/paper/5071-translating-embeddings-for-modeling-multi-relational-data.pdf

[4] Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. 2019. Meta relational learning for few-shot link prediction in knowledge graphs. *arXiv preprint arXiv:1909.01515* (2019).

[5] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017. Chains of Reasoning over Entities, Relations, and Text using Recurrent Neural Networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, 132–141. https://www.aclweb.org/anthology/E17-1013

[6] Zhengxiao Du, Chang Zhou, Ming Ding, Hongxia Yang, and Jie Tang. 2019. Cognitive Knowledge Graph Reasoning for One-shot Relational Learning. *CoRR* abs/1906.05489 (2019). arXiv:1906.05489 http://arxiv.org/abs/1906.05489

[7] Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. *CoRR* abs/1703.03400 (2017). arXiv:1703.03400 http://arxiv.org/abs/1703.03400

[8] Tianyu Gao, Xu Han, Zhiyuan Liu, and Maosong Sun. 2019. Hybrid Attention-Based Prototypical Networks for Noisy Few-Shot Relation Classification. In *AAAI*.

[9] Tianyu Gao, Xu Han, Ruobing Xie, Zhiyuan Liu, Fen Lin, Leyu Lin, and Maosong Sun. 2019. Neural Snowball for Few-Shot Relation Learning. *ArXiv* abs/1908.11007 (2019).

[10] Xu Han, Hao Zhu, Pengfei Yu, Ziyun Wang, Yuan Yao, Zhiyuan Liu, and Maosong Sun. 2018. FewRel: A Large-Scale Supervised Few-Shot Relation Classification Dataset with State-of-the-Art Evaluation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, 4803–4809. https://doi.org/10.18653/v1/D18-1514

[11] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[12] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. 2015. Siamese Neural Networks for One-shot Image Recognition.

[13] Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine learning* 81, 1 (2010), 53–67.

[14] J. Li, B. Chiu, S. Feng, and H. Wang. 5555. Few-Shot Named Entity Recognition via Meta-Learning. *IEEE Transactions on Knowledge and Data Engineering* 01 (nov 5555), 1–1. https://doi.org/10.1109/TKDE.2020.3038670

[15] Zhenguo Li, Fengwei Zhou, Fei Chen, and Hang Li. 2017. Meta-SGD: Learning to Learn Quickly for Few Shot Learning. *CoRR* abs/1707.09835 (2017). arXiv:1707.09835 http://arxiv.org/abs/1707.09835

[16] Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2019. Adapting Meta Knowledge Graph Information for Multi-Hop Reasoning over Few-Shot Relations. *arXiv preprint arXiv:1908.11513* (2019).

[17] Farzaneh Mahdisoltani, Joanna Biega, and Fabian M Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. (2015).

[18] T. Mitchell, W. Cohen, E. Hruschka, P. Talukdar, B. Yang, J. Betteridge, A. Carlson, B. Dalvi, M. Gardner, B. Kisiel, J. Krishnamurthy, N. Lao, K. Mazaitis, T. Mohamed, N. Nakashole, E. Platanios, A. Ritter, M. Samadi, B. Settles, R. Wang, D. Wijaya, A. Gupta, X. Chen, A. Saparov, M. Greaves, and J. Welling. 2018. Never-ending Learning. *Commun. ACM* 61, 5 (April 2018), 103–115. https://doi.org/10.1145/3191513

[19] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional Vector Space Models for Knowledge Base Completion. 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)*. Association for Computational Linguistics, Beijing, China, 156–166. https://doi.org/10.3115/v1/P15-1016

[20] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data.. In *ICML*, Vol. 11. 809–816.

[21] Meng Qu, Tianyu Gao, Louis-Pascal Xhonneux, and Jian Tang. 2020. Few-shot relation extraction via bayesian meta-learning on relation graphs. In *International Conference on Machine Learning*. PMLR, 7867–7876.

[22] Sachin Ravi and Hugo Larochelle. 2017. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*.

[23] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. 2016. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065* (2016).

[24] Jiawei Sheng, Shu Guo, Zhenyu Chen, Juwei Yue, Lihong Wang, Tingwen Liu, and Hongbo Xu. 2020. Adaptive Attentional Network for Few-Shot Knowledge Graph Completion. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 1681–1691. https://doi.org/10.18653/v1/2020.emnlp-main.131

[25] Jake Snell, Kevin Swersky, and Richard Zemel. 2017. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*. 4077–4087.

[26] 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*. https://openreview.net/forum?id=HkgEQnRqYQ

[27] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. *arXiv:1902.10197* (2019).

[28] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. 2018. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1199–1208.

[29] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*. 2071–2080.

[30] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, koray kavukcuoglu, and Daan Wierstra. [n.d.]. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems* (2016), D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (Eds.), Vol. 29. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2016/file/90e1357833654983612fb05e3ec9148c-Paper.pdf

[31] Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledge Base. *Commun. ACM* 57 (2014), 78–85. http://cacm.acm.org/magazines/2014/10/178785-wikidata/fulltext

[32] Hongwei Wang, Hongyu Ren, and Jure Leskovec. 2020. Entity Context and Relational Paths for Knowledge Graph Completion. *arXiv preprint arXiv:2002.06757* (2020).

[33] Weikang Wang, Jiajun Zhang, Qian Li, Chengqing Zong, and Zhifei Li. 2019. Are You for Real? Detecting Identity Fraud via Dialogue Interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 1762–1771. https://doi.org/10.18653/v1/D19-1185

[34] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*.

[35] Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2018. One-Shot Relational Learning for Knowledge Graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Association for Computational Linguistics, 1980–1990. https://aclanthology.info/papers/D18-1223/d18-1223

[36] Jinghui Yan, Yining Wang, Lu Xiang, Yu Zhou, and Chengqing Zong. 2020. A Knowledge-driven Generative Model for Multi-implication Chinese Medical Procedure Entity Normalization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*.

[37] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. In *International Conference on Learning Representations*.

[38] Zhi-Xiu Ye and Zhen-Hua Ling. 2019. Multi-Level Matching and Aggregation Network for Few-Shot Relation Classification. In *ACL*.

[39] Mo Yu, Xiaoxiao Guo, Jinfeng Yi, Shiyu Chang, Saloni Potdar, Yu Cheng, Gerald Tesauro, Haoyu Wang, and Bowen Zhou. 2018. Diverse Few-Shot Text Classification with Multiple Metrics. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, New Orleans, Louisiana, 1206–1215. https://doi.org/10.18653/v1/N18-1109

[40] Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V. Chawla. 2020. Few-Shot Knowledge Graph Completion. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 03 (April 2020), 3041–3048. https://doi.org/10.1609/aaai.v34i03.5698

[41] Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019. Long-tail Relation Extraction via Knowledge Graph Embeddings and Graph Convolution Networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics,

Minneapolis, Minnesota, 3016–3025. https://doi.org/10.18653/v1/N19-1306

[42] Yang Zhao, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2020. Knowledge Graphs Enhanced Neural Machine Translation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20.* 4039–4045.

[43] Chengqing Zong, Rui Xia, and Jiajun Zhang. 2021. *Text Data Mining.* Springer.