BiMetaR: A Bilateral Meta Relational Learning Model for Few-Shot Link Prediction in Knowledge Graphs

Jiawei Guan[1] , **Yuqi He**[2]
[1]Renmin University of China
[2]Renmin University of China
{2018202177, 2018202183}@ruc.edu.cn,

Abstract

Few-shot Knowledge Graph (KG) completion is a focus of current research, where each task aims querying unseen facts of a relation given its few-shot support entity pairs.Recent attempts solve this problem by learning representations of relations from its support entity pairs, ignoring their dynamic properties, i.e., support entity pairs may have different importance within task relations, and there may be some relations between the support pairs. This work proposes a novel network for relation embedding, based on a Meta Relational Learning framework. Our model utilized neural attention mechanism with Bidirectional Long Short-Term Memory Networks (BiLSTM) (Hochreiter and Schmidhuber, 1997). to extract the most important information of the relation from support pairs.

1 Introduction

A knowledge graph is composed by a large amount of triples in the form of (head entity, relation, tail entity) ((h, r, t)in short), encoding knowledge and facts in the world. Many KGs have been proposed and applied to various applications. Although with huge amount of entities, relations and triples, many KGs still suffer from incompleteness, thus knowledge graph completion is vital for the development of KGs. One of knowledge graph completion tasks is link prediction, predicting new triples based on existing ones. For link prediction, KG embedding methods(Bordes et al., 2013; Nickel et al., 2015; Trouillon et al., 2016; Yang et al., 2014)are promising ways. They learn latent representations, called embeddings, for entities and relations in continuous vector space and accomplish link prediction via calculation with embeddings.

However, those datasets (eg. FB15k, WN18) used by previous embedding methods mostly only cover common relations in KGs. For more practical scenarios, we believe the desired KG completion models should handle two key properties of KGs. First, a large portion of KG relations have very few instances.Therefore, it is crucial for models to be able to complete relations with limited numbers of triples. Second, to capture up-to-date knowledge, real world KGs are often dynamic and evolving at any given moment. New relations will be added whenever new knowledge is acquired. However, to predict target relations, previous methods usually rely on well-learned representations of these relations. In the dynamic scenario, the representations of new relations cannot be sufficiently trained given limited training instances, thus the ability to adapt to new relations is also limited for current models.

In contrast to previous methods, we devote to discuss few-shot link prediction in knowledge graphs, predicting tail entity t given head entity h and relation r by only observing K triples about r, usually K is small. We propose a new framework Bilateral Attentional Network for Meta Relational Learning (BiMetaR) for few-shot link prediction based on the intuition that the most important information to be transferred from a few exiting instances to incomplete triples should be the common and shared knowledge within one task.

2 Related Work

Recent years have seen increasing interest in learning representations for entities and relations in KGs, a.k.a KG embedding. Various methods have been devised, and roughly fall into three groups: 1) translation-based models which interpret relations as translating operations between head-tail entity pairs (Bordes et al., 2013); 2) simple semantic matching models which compute composite representations over entities and relations using linear mapping operations (Yang et al., 2014; Trouillon et al., 2016; Sun et al., 2019; Liu et al., 2017); and 3) (deep) neural network models which obtain composite representations using more complex operations (Schlichtkrull et al., 2018; Dettmers et al., 2018). Traditional embedding models always require sufficient training triples for all relations, thus are limited when sloving the few-shot problem.

Previous few-shot learning studies mainly focus on computer vision (Sung et al., 2018), imitation learning (Duan et al., 2017) and sentiment analysis (Li et al., 2019). Recent attempts (Xiong et al., 2018; Chen et al., 2019; Zhang et al., 2020; Sheng et al., 2020), tried to perform few-shot relational learning for long-tail relations. Xiong et al. Xiong et al. (2018) proposed a matching network GMatching, which is the first research on one-shot learning for KGs as far as we know. GMatching exploits a neighbor encoder to enhance entity embeddings from their one-hop neighbors,and uses a LSTM matching processor to perform a multi-step matching by a LSTM block. FSRL (Zhang et al., 2020) extends

GMatching to few-shot cases, further capturing local graph structures with an attention mechanism. Chen et al. Chen et al. (2019) proposed a novel meta relational learning framework MetaR by extracting and transferring shared knowledge across tasks from a few existing facts to incomplete ones. Sheng etal. Sheng et al. (2020) proposed an adaptive attentional network FAAN based on the dynamic properties of entities. However, previous studies learn entity and relations representations as the support pairs are not related and that they are equally important.

3 Task Formulation

In this section, we present the formal definition of a knowledge graph and few-shot link prediction task. A knowledge graph is defined as follows:

Definition 3.1 *(Knowledge Graph G) A knowledge graph G = $\{E, R, TP\}$. E is the entity set. R is the relation set. And $TP = \{(h, r, t) \in E \times R \times E\}$ is the triple set.*

And a few-shot link prediction task in knowledge graphs is defined as:

Definition 3.2 *(Few-shot link prediction task T) With a knowledge graph $G = \{E, R, TP\}$, given a support set $S_r = \{(h_i, t_i) \in E \times E | (h_i, r, t_i) \in TP\}$ about relation $r \in R$, where $|S_r| = K$, predicting the tail entity linked with relation r to head entity h_j, formulated as $r : (h_j, ?)$, is called K-shot link prediction.*

As defined above, a few-shot link prediction task is always defined for a specific relation. During prediction, there usually is more than one triple to be predicted, and with support set Sr , we call the set of all triples to be predicted as query set $Q_r = \{r : (h_j, ?)\}$. The goal of a few-shot link prediction method is to gain the capability of predicting new triples about a relation r with only observing a few triples about r. Thus its training process is based on a set of tasks $T_{train} = \{T_i\}_{i=1}^{M}$where each task $T_i = \{S_i, Q_i\}$ corresponds to an individual few-shot link prediction task with its own support and query set. Its testing process is conducted on a set of new tasks $T_{test} = \{T_j\}_{j=1}^{N}$ which is similar to T_{train}, other than that $T_j \in T_{test}$ should be about relations that have never been seen in T_{train}.

4 Our Approach

This section introduces our approach BiMetaR. Given a support set T_train, the purpose of BiMetaR is to extract relation meta and gradient meta and incorporate them with knowledge graph embedding to solve few-shot link prediction. Similar to the MetaR, BiMetaR consists of two major parts: (1) Relation-Meta Learner; (2) Embedding Learner.

The algorithm of BiMetaR is shown in Algorithmm 1. In each round of training, we first sample a support set and query sets from the training set in line 16, and obtain the corresponding entity representation in line 17. Then we use the BiLSTM model to calculate the representation of the relationship by calling the Get_Relation function from line 1 to 9. After that, we calculate loss, and update the parameters through two gradient descent back propagation, including the relation meta R, BiLSTM model parameters and entity embeddings.

Algorithm 1: Learning of BiMetaR

1 **Function** `Get_Relation`($\mathcal{T}_r$, emb, ϕ)**:**
2 Initialize(h_1, c_1, h_2, c_2)
3 **for** i *in* $[0, k]$ **do**
4 $h_1, c_1 = \phi_{forward}(\ emb[i], (h_1, c_1)\ )$
5 $h_2, c_2 = \phi_{reverse}(\ emb[i], (h_2, c_2)\ )$
6 $r1$.push_back(h_1 + $emb[i]$)
7 $r2$.push_back(h_2 + $emb[i]$)
8 R = Att((r_1 + r_2))
9 return R
10 **Function** `BiMetaR`($\mathcal{T}_{train}$, emb, ϕ, k)**:**
11 $\mathcal{T}_{train}$: Training tasks.
12 emb: Embedding Layer.
13 ϕ: Forward and Reverse LSTMCell.
14 k: number of entity pairs in support set.
15 **while** *Not Done* **do**
16 Sample a task $\mathcal{T}_r = (S_r, Q_r)$ from $\mathcal{T}_{train}$
17 Get corresponding embedding of entities emb_r
 R = Get_Relation ($\mathcal{T}_r$, emb_r, ϕ, k)
18 Compute loss in S_r
19 Get G by gradient of R
20 Update R by G
21 Compute loss in Q_r
22 Update ϕ and emb by loss in Q_r

4.1 Relation-Meta Learner

To extract the relation meta from support set, we design a relation-meta leaner to learn a mapping form head and tail entities in support set to relation meta. The structure of this relation-meta learner is implemented as a Bidirectional Long Short-Term Memory Networks (BiLSTM) with neural attention mechanism. Our model architecture is shown in Figure 1.

In task T_r, the input of relation-meta learner is head and tail entity pairs in support set $\{(h_i, t_i) \in S_r\}$. The input of t^{th} entity pair is shown in following equation:

$$x_t = h_t \oplus t_t \tag{1}$$

Typically, four components composite the LSTM-based recurrent neural networks: one input gate i_t with corresponding weight matrix W_{xi}, W_{hi}, W_{ci}, b_i; one forget gate f_t with corresponding weight matrix W_{xf} ,W_{hf} , W_{cf} , b_f ; one output gate o_t with corresponding weight matrix W_{xo}, W_{ho}, W_{co}, b_o, all of those gates are set to generate some degrees, using current input x_i, the state hi_{i-1} that previous step generated , and current state of this cell c_{i-1} (peephole), for the decisions whether to take the inputs, forget the memory stored before, and output the state generated later. Just as these following equations demonstrate:

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}hi_{t-1} + W_{ci}c_{t-1} + b_i\right) \tag{2}$$

$$f_t = \sigma\left(W_{xf}x_t + W_{hf}hi_{t-1} + W_{cf}c_{t-1} + b_f\right) \tag{3}$$

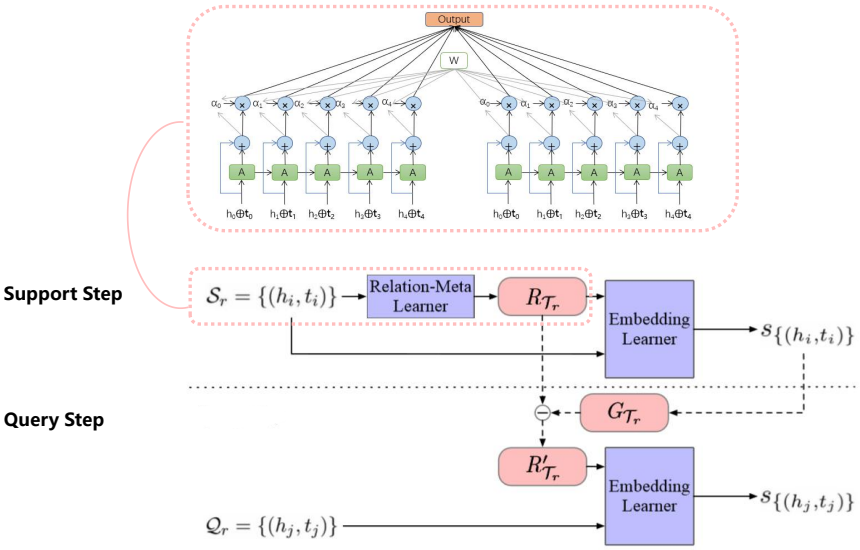

Figure 1: Overview of BiMetaR. $T_r = \{S_r, Q_r\}$, R_{Tr} and R'_{T_r} represent relation meta and updated relation meta, and G_{Tr} represents gradient meta

$$g_t = \tanh\left(W_{xc}x_t + W_{hc}hi_{t-1} + W_{cc}c_{t-1} + b_c\right) \quad (4)$$

$$c_t = i_t g_t + f_t c_{t-1} \quad (5)$$

$$o_t = \sigma\left(W_{xo}x_t + W_{ho}hi_{t-1} + W_{co}c_t + b_o\right) \quad (6)$$

$$hi_t = o_t \tanh\left(c_t\right) \quad (7)$$

The network contain two sub-networks for the left and right support pairs sequence, which are forward and backward pass respectively. The output of the i^{th} support pair is shown in the following equation:

$$R_{(h_i,t_i)} = \left[\overrightarrow{hi_i} \oplus \overleftarrow{hi_i}\right] \quad (8)$$

where we use element-wise sum to combine the forward and backword pass outputs.

With multiple relation meta, we generate the final relation meta in current task via neural attention network. Let H be a matric consisting of output vectors $[hi_1, hi_2, ..., hi_n]$ that the BiLSTM layer produced, where n is the length of current task set. The representation R_{T_r} of the task set is formed by a weighted sum of these output vectors:

$$M = tanh(H) \quad (9)$$

$$\alpha = softmax(\omega^T M) \quad (10)$$

$$R_{T_r} = H\alpha^T \quad (11)$$

4.2 Embedding Learner

As we want to get gradient meta to make a rapid update on relation meta, we need a score function to evaluate the truth value of entity pairs under specific relations and also the loss function for current task. However, few-shot task relations are always hard to obtain effective representations by existing embedding models that always require sufficient training data for the relations. Inspired by TransE (Bordes et al., 2013), we model the task relation embedding r as a translation between the entity embeddings h and t, i.e., we want $h + r \approx t$ when the triple holds. The intuition here originates from linguistic regularities such as Microsoft - Satya Nadella = Apple - Tim Cook, and such analogy holds because of the relation CEOof. Under the translation assumption, we can obtain the embedding of few-shot task entity t given its query pair (h,r):

$$\mathbf{t} = \mathbf{h} + \mathbf{r} \quad (12)$$

where r,t,$h \in R^d$; t and h are embeddings pretrained on G' with current embedding model such as TransE; d denotes the pre-trained embedding dimension.

In task T_r, we can calculate the score for each entity pairs (h_i, t_i) in support set S_r as follows:

$$s_{(h_i,t_i)} = \|\mathbf{h}_i + R_{\mathcal{T}_r} - \mathbf{t}_i\| \quad (13)$$

where $\|x\|$ represents the L2 norm of vector x.

with score function for each triple, we set the following loss,

$$L\left(\mathcal{S}_r\right) = \sum_{(h_i,t_i)\in\mathcal{S}_r} \left[\gamma + s_{(h_i,t_i)} - s_{\left(h_i,t'_i\right)}\right]_+ \quad (14)$$

where $[x]_+$ represents the positive part of x and γ represents margin which is a hyperparameter. $s_{\left(h_i,t'_i\right)}$ is the score for

negative sample. (h_i, t_i') corresponding to current positive entity pair $(h_i, t_i) \in \mathcal{S}_r$, where $(h_i, r, t_i') \notin G$.

$L(\mathcal{S}_r)$ should be small for task T_r which represents the model can properly encode truth values of triples. Thus gradients of parameters indicate how should the parameters be updated. Thus we regard the gradient of R_{T_r} based on $L(\mathcal{S}_r)$ as gradient meta G_{T_r} :

$$G_{\mathcal{T}_r} = \nabla_{R_{\tau_r}} L(\mathcal{S}_r) \tag{15}$$

Following the gradient update rule, we make a rapid update on relation meta as follows:

$$R'_{\mathcal{T}_r} = R_{\mathcal{T}_r} - \beta G_{\mathcal{T}_r} \tag{16}$$

where β indicates the step size of gradient meta when operating on relation meta.

When scoring the query set by embedding learner, we use updated relation meta. After getting the updated relation meta R', we transfer it to samples in query set $Q_r = \{(h_j, t_j)\}$ and calculate their scores and loss of query set, following the same way in support set.

4.3 Training Objective

During training, our objective is to minimize the following loss L which is the sum of query loss for all tasks in one minibatch:

$$L = \sum_{(\mathcal{S}_r, \mathcal{Q}_r) \in \mathcal{T}_{\text{train}}} L(\mathcal{Q}_r) \tag{17}$$

5 Experiments

In this section, we measure the performance of our BiMetaR method and compare it with 2 baselines for evaluation.

5.1 Experimental Setup

We exhibit our evaluation setup in this part.

Dataset. We use dataset NELL in our evaluation, which is constructed by Xiong et al. Furthermore, as this benchmark is first used in GMatching which consider both learned embeddings and one-hop graph structures, a background graph is constructed with relations out of training/validation/test sets for obtaining the pre-train entity embeddings and providing the local graph for GMatching. The previous MetaR model itself does not require background graph data, but it makes good use of it to sample data by fitting it into training tasks or using it to train embeddings to initialize entity representations, thereby obtaining good benefits. In addition, MetaR also supports a training method that samples both from background graph and the training data.

Metrics. We use two traditional metrics to evaluate different methods on NELL, MRR and Hits@k. MRR is the mean reciprocal rank and Hits@k is the proportion of correct entities ranked in the top k in link prediction. The k is set to 1, 5, and 10. The few-shot size is set to 5 for the following experiments.

Baselines. We consider two categories of baseline methods for comparison:

- Relational Embedding Method. This type of model learns entity/relation embeddings by modeling relational structure in KG. All entity pairs of background relations and training relations, as well as few-shot training entity pairs of validate and test relations are used to train models. We employ MetaR (Chen et al., 2019) and FAAN (Sheng et al., 2020) for comparison.
- Graph neighbor encoder methods. This type of model joints graph local neighbor encoder and matching network to learn entity embeddings and predict facts of new relations. We employ state-of-the-art model GMatching (Xiong et al. 2018) for comparison.

Platform. The CPU used in our evaluation is Intel(R) Xeon(R) E5-2620 v3, the GPU is GerForce GTX TIT. The operating system is Ubuntu 16.04.

5.2 Implementation

We use mini-batch gradient descent with size set as 64 for our training. We use Adam (Kingma and Ba, 2015) with the initial learning rate as 0.001 to update parameters. We set = 1 and = 1. The number of positive and negative triples in query set is 3. Trained model will be applied on validation tasks each 1000 epochs, and the current model parameters and corresponding performance will be recorded, after stopping, the model that has the best performance on MRR will be treated as final model. For number of training epoch, we use early stopping with 30 patient epochs, which means that we stop the training when the performance on MRR drops 30 times continuously. Following GMatching, the embedding dimension of NELL is 100.

5.3 Evaluation Method and Results Comparison

We modified different models according to the flexibility of their methods, and tested the results under different conditions.

For GMatching, we tested the results using different pre-training methods. We employed two widely used methods: TransE (Bordes et al., 2013) and ComplEx (Trouillon et al., 2016) for processing the pre-trained embedding representation. The first two rows of Table 1 shows the running results of Gmatching under these two preprocessing methods. The results show that all metrics using CompIE are better than TransE.

For FAAN, we tested the influence of attention mechanism and the results are shown in line 3-4 of Table 1. Obviously the attention mechanism has brought great benefits to the model.

Inspired by MetaR's training method, we incorporate the background graph data and pre-trained entity representation into training process. In our experimental results in line 5 to 6, adding background image data will bring great performance gains.

Besides, we also tried using two different encoders to change the entity encoding rules of the MetaR model. On the basis of loading the pre-trained embeddings, we first try to use the average value of one-hop neighbors as the embedding of the entity. We also use the attention model to represent entities based on the relationship between the node and the one-hop neighbor and the similarity between the query relationship. Results of above two methods is shown in line 7-8 in Table 1.

Table 1: Overall results of all methods.

Model	Hits@1	Hits@5	Hits@10	MRR
GMatching(TransE)	0.09	0.170	0.198	0.159
GMatching(CompIE)	0.108	0.177	0.212	0.167
FAAN	0.172	0.277	0.316	0.223
FAAN(Attention)	0.185	0.275	0.321	0.231
MetaR	0.160	0.301	0.363	0.231
MetaR(Background Graph)	0.197	0.412	0.502	0.302
MetaR(OneHop)	0.166	0.354	0.439	0.260
MetaR(Adaptive Neighbor)	0.147	0.310	0.380	0.224
MetaR(LSTM)	0.193	**0.424**	**0.507**	0.300
BiMetaR	**0.202**	0.411	0.503	**0.306**

In order to model the influence of support set entities on the relationship representation, we first implemented a LSTM with residual and attention mechanism to improve the MetaR model. Then, considering that LSTM is more suitable for natural language processing tasks where there is a sequence dependency between data, we change LSTM to BiLSTM to reduce the impact of order. Our approach brings a promising performance improvement. Specifically, for 1-shot link prediction, BiMetaR increases by 26% and 32% on Hits@1 and MRR compared with MetaR method(without using background graph).

6 Conclusion

In this paper, we proposed a Bilateral Meta Relational Learning framework based on MetaR to do few-shot link prediction in KGs, and we design our model to transfer relation-specific meta information from support set to query set. BiMetaR performs joint optimization of considering different weight between entity pairs in support set and the query set. The extensive experiments on dataset NELL demonstrate that BiMetaR can outperform state-of-the-art baseline methods, and it is also independent with background knowledge graphs.

References

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*, pages 1–9, 2013.

Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104 (1):11–33, 2015.

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

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

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

Hanxiao Liu, Yuexin Wu, and Yiming Yang. Analogical inference for multi-relational embeddings. In *International conference on machine learning*, pages 2168–2178. PMLR, 2017.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

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

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1199–1208, 2018.

Yan Duan, Marcin Andrychowicz, Bradly C Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-shot imitation learning. *arXiv preprint arXiv:1703.07326*, 2017.

Zheng Li, Xin Li, Ying Wei, Lidong Bing, Yu Zhang, and Qiang Yang. Transferable end-to-end aspect-based sentiment analysis with selective adversarial learning. *arXiv preprint arXiv:1910.14192*, 2019.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. One-shot relational learning for knowledge graphs. *arXiv preprint arXiv:1808.09040*, 2018.

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

Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V Chawla. Few-shot knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3041–3048, 2020.

Jiawei Sheng, Shu Guo, Zhenyu Chen, Juwei Yue, Lihong Wang, Tingwen Liu, and Hongbo Xu. Adaptive attentional network for few-shot knowledge graph completion. *arXiv preprint arXiv:2010.09638*, 2020.