HMNet: Hybrid Matching Network for Few-Shot Link Prediction

Shan Xiao[1], Lei Duan[1](✉), Guicai Xie[1], Renhao Li[1], Zihao Chen[1], Geng Deng[1], and Jyrki Nummenmaa[2]

[1] School of Computer Science, Sichuan University, Chengdu, China
{shanxiao,guicaixie,lirenhao,chenzihao}@stu.scu.edu.cn, leiduan@scu.edu.cn

[2] Tampere University, Tampere, Finland
jyrki.nummenmaa@tuni.fi

Abstract. Knowledge graphs (KGs) are widely used in many real-world applications, such as information retrieval, question answering system, and personal recommendation. However, most KGs are suffering from the incompleteness problem. To deal with the task of link prediction, previous knowledge graph embedding methods require numerous reference instances for each relation. It is worth noting that most relations in KGs have only a few reference instances available. Existing works for few-shot link prediction evaluate the authenticity of triplets from a single relation perspective. In this paper, we propose Hybrid Matching Network (HMNet) for few-shot link prediction, evaluating triplets from entity and relation two perspectives. At the entity-aware matching network, HMNet uses attentive inductive embedding layer to aggregate entity features and relation-aware topology, and then provides entity-aware score to implement first perspective evaluation. At the relation-aware matching network, HMNet integrates feature attention mechanism to implement relation perspective evaluation. Experiments on two public datasets indicate that HMNet achieves promising performance in few-shot link prediction.

Keywords: Few-shot link prediction · Hybrid matching network · Feature attention mechanism

1 Introduction

Knowledge graphs (KGs), collection of triplets (e.g., <*head entity, relation, tail entity*>), have been widely used in a range of applications, such as question answering [3], recommender system [29], and information retrieval [5]. A typical large-scale KG, such as Freebase [1] or YAGO [18], contains billions of triplets. However, they are suffering from the incompleteness problem [26]. For instance, 75% of person entities have no nationality information in Freebase [7]. As a

This work was supported in part by the National Natural Science Foundation of China (61972268), the Sichuan Science and Technology Program (2020YFG0034), and the Academy of Finland (327352).

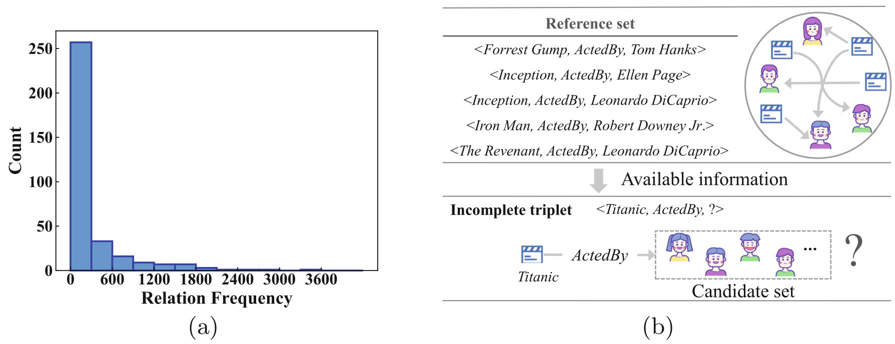


Fig. 1. (a) The histogram of relation frequencies in NELL-One dataset; and (b) Illustration of an example of few-shot link prediction (Each edge denotes a reference instance).

result, it is hard to accurately answer questions like "How many users are from the same country as *Leonardo DiCaprio*?".

Clearly, it is time-consuming and labor intensive to deal with all incomplete triplets manually. Thus automatically completing a knowledge graph, which is also called *link prediction*, has become an important research task. There are some studies [2,6,15,16,20,27] that have been proposed to predict missing values in incomplete triplets based on existing knowledge. The main idea of these methods is consummating incomplete triplets via low-dimensional representations of entities and relations. However, the precondition of these methods solving the incompleteness problem is that each relation contains numerous reference instances.

It is worth noting that the relation frequency in many KGs always shows a long-tail distribution, as shown in Fig. 1(a). That is, a large portion of relations have only a few reference instances [26]. The automatic completion of knowledge graph under long-tail distribution is called few-shot link prediction task. In this task, only a few reference instances are available for each relation. To better understand the task, an example is shown in Fig. 1(b).

Example 1. In an existing KG, we need to perform link prediction on relation "*ActedBy*", that is, predicting tail entity from the candidate set given head entity "*Titanic*" and relation "*ActedBy*". Different from other relations, "*ActedBy*" has only 5 reference instances. Therefore, there is a little information available for it. Few-shot link prediction task focuses on these less informativeness relations.

There are some few-shot learning studies [4,26,28] for the above task. Specifically, these methods first construct query triplets by splicing all candidate tail entities with the given head entity, and then learn embeddings of query triplets and representation of the relation of reference set. Finally, they match each query triplet with the reference set to get a score. The goal is to make true triplets rank high. Obviously, for each query triplet, these methods evaluate its authenticity from a single relation perspective. In addition, most of existing works calculate the score for each triplet by applying dot product under the assumption that all features of the relation contribute equally. Although these methods achieve encouraging improvements, the performance remains unsatisfactory.

In this paper, we propose a novel model Hybrid Matching Network (HMNet) for few-shot link prediction. It consists of an entity-aware matching network and a relation-aware matching network. HMNet can evaluate the authenticity of triplets from two perspectives:

- *Entity perspective*: The entity-aware matching network obtains the entity-aware scores between different candidates and reference instances as the first perspective evaluation.
- *Relation perspective*: The relation-aware matching network obtains the relation-aware scores as the second perspective. It simultaneously weights different features of relation with unequal contributions when calculating the score.

The final prediction result of each triplet is acquired by combining these two matching scores.

The contributions of this work are summarized as follows:

- designing a novel model HMNet, which employs hybrid matching and integrates attention mechanism for few-shot link prediction.
- pointing out the importance of entity-aware matching, and providing an extra perspective evaluation.
- evaluating HMNet model on two public datasets. Empirical results prove the effectiveness of our proposed model HMNet over many competitive baselines.

The rest of the paper is organized as follows. We review related work in Sect. 2, and formulate the problem of few-shot link prediction in Sect. 3. In Sect. 4, we discuss the critical techniques of the proposed model HMNet. We report a systematic empirical evaluation in Sect. 5, and conclude the paper in Sect. 6.

2 Related Work

Our work is related to the existing research on knowledge graph embedding and few-shot learning. We introduce the related work briefly below.

2.1 Knowledge Graph Embedding

To consummate incomplete triplets in KGs, it is vital to obtain embeddings of entities and relations in the continuous low-dimensional space. Existing knowledge graph embedding models can be divided into two main categories: distance based models and bilinear based models.

Aiming to translate distance between entity pairs, Bordes *et al.* [2] first proposed a translational distance based method TransE. It can obtain low-dimensional embeddings by optimizing the distance function between triplets of the relational semantic. After that, in order to break through the limitation of TransE in dealing with complex relations, several models have been proposed, such as TransH (Wang *et al.* [24]) and TransR (Lin *et al.* [13]). Focused on tensor decomposition, Nickel *et al.* [15] firstly designed a bilinear model RESCAL, which can obtain relation embeddings by modeling the potential structure of KGs. Later, DistMult proposed by Yang *et al.* [27] simplifies RESCAL by limiting the relational matrix to the diagonal matrix. ComplEx introduced by Trouillon *et al.* [20] extends DistMult into the complex space to better model reversible relations in KGs.

The performance of above methods strongly relies on numerous reference instances. In practical applications, these methods fail to achieve their expected performance, due to the relation frequency in real datasets often has a long-tail distribution.

2.2 Few-Shot Learning

Few-shot learning enables models to achieve impressive results with insufficient data. Existing approaches include learning a metric space over input features [12,17,21,26], such that similar instances are close together while dissimilar can be more easily differentiated. Recently, meta-learning is proposed to solve few-shot learning problem. Specially, the meta-learner gradually learns generic information (meta-knowledge) across tasks, and task-learner generalizes to the new task based on meta-knowledge and specific information of the new task [8,14,23]. Although few-shot learning has developed fast in recent years, it mainly focus on computer vision applications and text classification.

To the best of our knowledge, the work proposed by Xiong *et al.* [26] is the first research on few-shot link prediction. It's a metric based model called GMatching, which includes two components: neighbor encoder and matching processor. The neighbor encoder uses entities' one-hop neighbors to obtain their embeddings. And then each relation representation is obtained by concatenating the embeddings of the head entity and tail entity. The matching processor matches each query instance with the reference set. Following the work of GMatching [26], Zhang *et al.* [28] proposed a relation-aware heterogeneous neighbor encoder based on the attention mechanism to learn entity embedding, and used recurrent auto-encoder to aggregate information from reference instances. Chen *et al.* [4] employed the relation-specific meta information transferring from the reference set to query set and proposed the MetaR model.

Previous methods solve the few-shot link prediction task only considering the relation perspective evaluation. This work is attempting to design a new framework that can evaluate the authenticity of triplets from two perspectives by leveraging valuable semantic information provided by the reference set.

3 Problem Definition

We start with some preliminaries. Let $\mathcal{E}$ and $\mathcal{R}$ be the sets of entities and relations, respectively. A knowledge graph is viewed as a graph $\mathcal{G} = \{(h, r, t)\} \subseteq$

$\mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $h \in \mathcal{E}$ and $t \in \mathcal{E}$ represent the head entity and tail entity, respectively, and $r \in \mathcal{R}$ denotes a specific relation connecting h and t. The goal of link prediction is to predict the missing values in incomplete triplets when two elements are given. In this study, we focus on predicting the tail entity given the head entity and query relation.

Under the few-shot learning setting, the model can be optimized on the set of training tasks $\mathcal{T}_{train} = \{\mathcal{T}_i\}_{i=1}^{M}$ and its generalization can be evaluated on the set of test tasks $\mathcal{T}_{test} = \{\mathcal{T}_j\}_{j=1}^{N}$. Each task $\mathcal{T}_i = \left\{\mathcal{D}^{ref}, \mathcal{D}^{query}\right\}$ corresponds to a few-shot learning task with reference set $\mathcal{D}^{ref}$ and query set $\mathcal{D}^{query}$. Each task $\mathcal{T}_j \in \mathcal{T}_{test}$ is similar to $\mathcal{T}_i$. According to the reference instances, the model needs to make prediction for instances in the query set. It should be noted that all tasks in testing are invisible in training, that is, $\mathcal{T}_{train} \cap \mathcal{T}_{test} = \varnothing$.

Definition 1 (Few-shot link prediction). *Few-shot link prediction is defined as a task to predict the true tail entity t_j of the missing triplet $(h_j, r, ?)$, given the reference set $\mathcal{D}_r^{ref} = \{(h_i, t_i) \mid (h_i, r, t_i) \in \mathcal{G}\}$ of relation r. $K = |\mathcal{D}_r^{ref}|$ represents the number of triplets in reference set, which is a small number. The set of all instances to be predicted of relation r is the query set $\mathcal{D}_r^{query} = \left\{(h_j, c_j)|c_j \in C_{h_j,r}\right\}$, where $C_{h_j,r}$ is candidate tail entities set for a given head entity h_j and relation r ($C_{h_j,r}$ including the true tail entity t_j).*

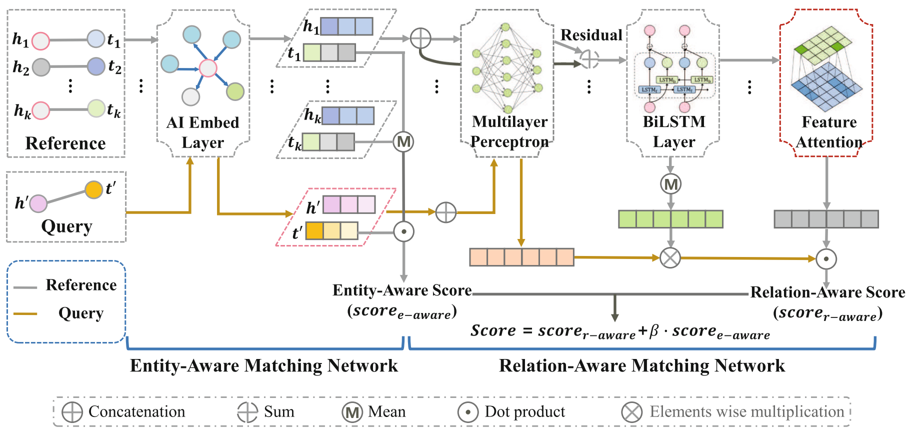
In the few-shot link prediction task, $\mathcal{R}_1$ and $\mathcal{R}_2$ are sets of relations involved in training and testing, respectively, and $\mathcal{R}_1 \cap \mathcal{R}_2 = \varnothing$. Each task corresponds to a relation $r \in \mathcal{R}_1 \cup \mathcal{R}_2$. Following the standard problem definition of work [26], we assume that the method to solve the task can access a background graph $\mathcal{G}'$, where $\mathcal{G}' = \{(h, r, t)|(h, r, t) \in \mathcal{G} \land r \in \mathcal{R} \setminus (\mathcal{R}_1 \cup \mathcal{R}_2)\}$.

4 The Design of HMNet

In this section, we present the details of HMNet. Figure 2 shows the framework of HMNet, which includes two components: entity-aware matching network and relation-aware matching network. Different from previous studies that focus on single relation perspective evaluation, HMNet can evaluate the authenticity of triplets from two perspectives. In Sect. 4.1, we describe the mechanism of entity-aware matching network. Relation-aware matching network for evaluation is described in Sect. 4.2.

4.1 Entity-Aware Matching Network

The objective of entity-aware matching network is to evaluate triplets from entity perspective. Specifically, it assigns high scores for true tail entities of triplets with tail entities' information in the reference set. This component consists of following two modules: attentive inductive embedding layer and entity-aware score.



h_1 t_1 h_2 t_2 h_k t_k
Reference
Query
h' t'
AI Embed Layer
Multilayer Perceptron
Residual
BiLSTM Layer
Feature Attention
Entity-Aware Score ($score_{e-aware}$)
Relation-Aware Score ($score_{r-aware}$)
$Score = score_{r-aware} + \beta \cdot score_{e-aware}$
Entity-Aware Matching Network
Relation-Aware Matching Network
Concatenation
Sum
Mean
Dot product
Elements wise multiplication

Fig. 2. Illustration of the proposed HMNet model.

Attentive Inductive Embedding Layer (AI Embed Layer). Low dimensional representations of nodes in the network have been proved useful in a variety of graph analysis tasks [10]. Existing works show that it is beneficial to use the relation-aware topology of an entity for link prediction task [26,30]. In addition, attention mechanism is widely used in recent deep learning studies [25,28]. Different from existing few-shot link prediction works [26,28] only modeling the relation-aware topology explicitly, HMNet employs AI Embed Layer to obtain entity embedding. Since it simultaneously captures the relation-aware topology and entity features, AI Embed Layer retains the advantages of previous methods and fully aggregates the information provided by reference set.

Specifically, for any entity e, the set of link information with head entity e in $\mathcal{G}'$ denotes as $\mathcal{I}_e = \{(r,t)|(e,r,t) \in \mathcal{G}'\}$. Hence, entity e is assigned relation-aware topology embedding as follows,

$$\begin{aligned} \mathcal{F}_e &= \sigma\left(\sum\nolimits_{(r,t)\in\mathcal{I}_e} a_{(r,t)}(\mathbf{W_1}[\boldsymbol{v}_r \oplus \boldsymbol{v}_t])\right) \\ a_{(r,t)} &= \frac{exp(\mathbf{P}(\mathbf{W_1}[\boldsymbol{v}_r \oplus \boldsymbol{v}_t]))}{\sum_{(r',t')\in\mathcal{I}_e} exp(\mathbf{P}(\mathbf{W_1}[\boldsymbol{v}_{r'} \oplus \boldsymbol{v}_{t'}]))} \end{aligned} \tag{1}$$

where $\oplus$ represents concatenation operation, $\boldsymbol{v}_r \in \mathbb{R}^d$ and $\boldsymbol{v}_t \in \mathbb{R}^d$ are pretrained embeddings of the relation r and tail entity t, respectively. d is the embedding size. σ represents the $Tanh$ activation function, and $a_{(r,t)}$ indicates the weight of link information (r,t) when representing the entity e. $\mathbf{P} \in \mathbb{R}^{1\times d}$ and $\mathbf{W_1} \in \mathbb{R}^{d\times 2d}$ are trainable weight matrices.

Aggregating features of entity e with its relation-aware topology representation has been widely used in many tasks and achieves good performance [22,30]. In order to make full use of the information of reference set, AI Embed Layer further combines them to get the entity embedding of e: $\boldsymbol{\omega}_e = \mathbf{W_2}\boldsymbol{v}_e + \mathcal{F}_e$, where $\boldsymbol{v}_e$ is the pre-trained embedding of e and $\mathbf{W_2} \in \mathbb{R}^{d\times d}$ is a trainable weight matrix.

Entity-Aware Score. The problem we tackle is that given a *head entity* and a *relation*, we need to predict the *tail entity*. According to the information of tail entities in the reference set, for each query instance (h', t'), we can calculate the entity-aware score to implement the first perspective evaluation. First, by applying the AI Embed Layer to each tail entity t from $\mathcal{D}_r^{ref}$ and (h', t'), HMNet gets the representation $\boldsymbol{\omega}_t$ of t. Then, HMNet summarizes output features of tail entities in the reference set $\mathcal{D}_r^{ref}$ as follows,

$$\boldsymbol{\mathcal{E}}_{ref} = \frac{1}{K}\sum\nolimits_{i=1}^{K} \boldsymbol{\omega}_{t_i} \tag{2}$$

where $t_i \in \left\{t|(h,t) \in \mathcal{D}_r^{ref}\right\}$. Finally, HMNet calculates the entity-aware score for (h', t'). Without loss of generality, HMNet employs the following way to calculate entity-aware score,

$$score_{e-aware} = \boldsymbol{\mathcal{E}}_{ref} \odot \boldsymbol{\omega}_{t'} \tag{3}$$

where $\odot$ represents dot product.

4.2 Relation-Aware Matching Network

To implement the relation perspective evaluation, we design a relation-aware matching network. In this section, we start by describing how to get the embedding of corresponding relation based on reference set. And then we discuss how to select more discriminative features to achieve more appropriate relation perspective evaluation.

Relation Encoder. HMNet assumes that each query instance expresses a special relation, and then measures whether this relation is similar to the relation expressed by reference set. HMNet employs the multilayer perceptron to encoder entity pairs. It can obtain the relation embedding represented by entity pair (h, t) as follows,

$$\boldsymbol{e}_{r\leftarrow(h,t)} = \mathbf{W}_r(\mathbf{W}[\boldsymbol{\omega}_h \oplus \boldsymbol{\omega}_t]) + [\boldsymbol{\omega}_h \oplus \boldsymbol{\omega}_t] \tag{4}$$

where $\boldsymbol{\omega}_h$ and $\boldsymbol{\omega}_t$ are embeddings of the head entity and tail entity, respectively, obtained by applying AI Embed Layer to h and t. $\mathbf{W}_{\boldsymbol{r}} \in \mathbb{R}^{2d\times 4d}$ and $\mathbf{W} \in \mathbb{R}^{4d\times 2d}$ are trainable weight matrices.

If the relation representation of each instance in $\mathcal{D}_r^{ref}$ is far away from each other, the resulting prototype vector of relation cannot capture common and representative features. Here, we employ a network to perform information propagation between reference instances, so that reference instances are closer in the metric space. LSTM network [11] has achieved good performance in the NLP field based on the long-distance information memory characteristic. But it can

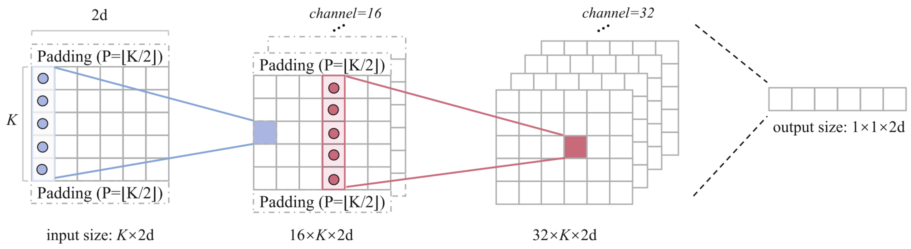


Fig. 3. The architecture of feature attention module.

only achieve unidirectional information propagation. Therefore, HMNet uses the BiLSTM network to implement bidirectional information propagation.

Given relation representations of reference instances, calculated by Eq. 4, BiLSTM performs information propagation on them to obtain the new relation representation $\boldsymbol{s}_r(h_i, t_i)$ for each reference instance (h_i, t_i): $\boldsymbol{s}_r(h_i, t_i) = \text{BiLSTM}(\boldsymbol{e}_{r \leftarrow (h_i, t_i)})$.

Relation-Aware Score. The works by Xiong *et al.* [26] and Zhang *et al.* [28] use dot product to calculate the score for each query instance. These methods believe that all features of relation contribute equally. However, when few-shot reference instances are used to represent the corresponding relation, the obtained information is limited and may contain noise information. It is hard to accurately capture all unique features of the relation. Therefore, we should pay more attention to discriminative features. HMNet is required to measure the importance of captured features when calculating score under the few-shot setting.

Since there are only a few reference instances for each relation, it is difficult to extract important features using feature engineering algorithms. Inspired by the work of [9] on text classification, HMNet uses a feature attention module to measure the importance of relation features, which enhances generality applicability of the model.

The feature attention module uses the convolution operation to iteratively update feature weights. The relation representation of each instance in the reference set is combined to form a matrix $\boldsymbol{s}_r^K \in \mathbb{R}^{K \times 2d}$. Figure 3 shows the module framework.

In order to aggregate the information of reference set when measuring the importance of each feature, the size of all convolution kernels of this module is set to $K \times 1$. More specific steps are as follows:

Step 1: HMNet uses 16 convolution kernels to perform convolution operation on $\boldsymbol{s}_r^K$, and sets stride to 1×1. It pads the bias to participate in the calculation. The output $\boldsymbol{H}_r^K \in \mathbb{R}^{16 \times K \times 2d}$ can be obtained.

Step 2: HMNet uses 32 convolution kernels to perform convolution operation again. Channels are 16, other settings are the same as previous step. The output is $\boldsymbol{H}_r^K \in \mathbb{R}^{32 \times K \times 2d}$.

Step 3: A convolution kernel with 32 channels is used, and stride is set to $K \times 1$ to obtain feature attention weight $O_r^K \in \mathbb{R}^{1\times1\times2d}$ for relation matching.

Algorithm 1. HMNet

```
Input: Training task set $\mathcal{T}_{train}$; Background graph $\mathcal{G}'$; The number of training steps $N_{step}$; Learning rate $\alpha$; Margin distance $\gamma$; Hyperparameter $\beta$
Output: $\theta$: Learning parameters of HMNet
 1: Load the pre-trained embeddings;
 2: Initialize $\theta$;
 3: for $i = 1 \rightarrow N_{step}$ do
 4:     Shuffle the tasks in $\mathcal{T}_{train}$;
 5:     for each task $\mathcal{T}_r$ in $\mathcal{T}_{train}$ do
 6:         Sample reference set $\mathcal{D}_r^{ref}$ and positive query instances set $\mathcal{Q}_r$ from $\mathcal{T}_r$;
 7:         Construct negative query instances set $\mathcal{Q}_r^-$ by replacing tail entities of $\mathcal{Q}_r$;
 8:         Compute the matching score for each triplet in $\mathcal{Q}_r \cup \mathcal{Q}_r^-$ using Eq.6;
 9:         Compute the loss $\mathcal{L}$ using Eq. 7;
10:         $\theta \leftarrow \mathrm{Adam}(\nabla_\theta \mathcal{L}, \theta, \alpha, \beta)$;
11:     end for
12: end for
13: return $\theta$;
```

After applying the relation encoder module, we obtain the relation representation of each reference instance in a metric space. Taking these representations as input of the feature attention module, HMNet gets feature attention weights. Like the tail entity information aggregation, HMNet uses the average of representations of reference instances to get the prototype representation of relation r: $\boldsymbol{c}_r = \frac{1}{K}\sum_{i=1}^{K} \boldsymbol{s}_r(h_i, t_i)$. Then HMNet reduces the dimensionality of the feature attention weights, so that $O_r^K \in \mathbb{R}^{1\times1\times2d} \rightarrow O_r^K \in \mathbb{R}^{2d}$. For each query instance (h', t'), HMNet uses O_r^K to calculate the final relation-aware score:

$$score_{r-aware} = O_r^K \odot (\boldsymbol{c}_r \otimes \boldsymbol{e}_{r\leftarrow(h',t')}) \tag{5}$$

where $\otimes$ denotes elements wise multiplication, and $\boldsymbol{e}_{r\leftarrow(h',t')}$ represents the relation embedding between h' and t' which is calculated by Eq. 4.

For a query instance (h', t'), the final score is:

$$Score = score_{r-aware} + \beta \cdot score_{e-aware} \tag{6}$$

where β is the hyperparameter indicating the weight of entity perspective evaluation.

4.3 Learning Objective and Algorithm

Given the background graph $\mathcal{G}'$, and reference set $\mathcal{D}_r^{ref}$ of relation r, we aim to select the true tail entity t_j from the candidate set for each h_j. Based on this learning objective, we rewrite the loss function following the definition of [28],

$$\mathcal{L}_\theta = \sum_{r\in\mathcal{R}_1} \sum_{(h_j,t_j)\in\mathcal{Q}_r} \sum_{(h_j,t_j^-)\in\mathcal{Q}_r^-} [\gamma - Score_{(h_j,t_j)} + Score_{(h_j,t_j^-)}]_+ \tag{7}$$

where $\mathcal{Q}_r = \{(h_j, t_j)|(h_j, r, t_j) \in \mathcal{G}\}$ is the set of positive query instances of relation r, $\mathcal{Q}_r^- = \left\{(h_j, t_j^-)|(h_j, r, t_j^-) \notin \mathcal{G}\right\}$ is the set of negative query instances of relation r, which is constructed by replacing tail entities of positive instances. $\mathcal{L}_\theta$ is standard hinge loss, and γ is margin distance.

Based on the discussions above, we present the pseudo-code of HMNet in Algorithm 1.

5 Experiments

We evaluate the performance of HMNet on two public datasets. All experiments are conducted on a server with an RTX2080 Ti and 11 GB memory. The model HMNet is implemented by Python 3.6 based on Pytorch 1.5.1.

5.1 Experimental Setup

Datasets: 1) NELL-One[1] consists of 181,109 triplets, 68,545 entities, and 358 relations. 2) Wiki-One, which is a subset of Wikidata[2], consists of 5,829,240 triplets, 4,838,244 entities, and 822 relations. Following the experimental settings of work [26], we select relations with less than 500 but more than 50 triplets as few-shot link prediction tasks. Table 1 shows the statistics of two datasets (#Training/Validation/Test denotes the number of relations for training/validation/testing).

Table 1. Statistics of the datasets.

Dataset	#Entities	#Relations	#Triplets	#Training/Validation/Test
NELL-One	68,545	358	181,109	51/5/11
Wiki-One	4,838,244	822	5,829,240	133/16/34

Baselines: In our experiments, several related methods are selected as baselines.

- **Knowledge graph embedding methods.** Knowledge graph embedding methods map relations and entities into continuous low-dimensional space. **TransE** [2] is a translational distance based method which defines the score function as $f_r(h, t) = -\|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|_{1/2}$. **RESCAL** [15] is a bilinear based method. This method represents each relation as a full rank matrix $\boldsymbol{M}_r$. **DistMult** [27] uses a bilinear score function to compute scores of knowledge triplets. **ComplEx** [20] extends DistMult to the complex space instead of real-valued ones.

[1] http://rtw.ml.cmu.edu/rtw/.
[2] https://test.wikidata.org.

– **Few-shot learning methods.** These models use a background graph $\mathcal{G}'$ to get the pre-trained embeddings of entities and learn a representation of relation. Then they adopt different score functions to get the ranking. **GMatching** [26] tackles the problem by enhancing the representation of entity and learning a relation metric space. **MetaR** [4] proposes relation-meta and gradient-meta two kinds of relation-specific meta information to solve this problem. **FSRL** [28] extends GMatching [26], from one-shot link prediction to few-shot link prediction.

Evaluation Metrics: Two metrics Hits@k and MRR are applied to evaluate the performance of the proposed model. Hits@k is the proportion of the correct tail entities in the top-k of all candidate entities. MRR (Mean Reciprocal Rank) is the average of all correct tail entities reciprocal ranking.

Implementation Details: For TransE, ComplEx, and DistMult, the implementation[3] released by Sun *et al.* [19] is adopted in our experiments. For RESCAL, we implement it by ourselves. For the above knowledge graph embedding methods, all the triplets from $\mathcal{G}'$ and training set are utilized for training. In addition, for each relation, K triplets from validation and test sets are chosen for training. In iterative training, only one negative sample is constructed for each true triplet in the batch task by replacing tail entity. Following GMatching [26], the embedding dimension is set to 100 and 50 for NELL-One and Wiki-One datasets, respectively. The maximum number of neighbors is set to 50 and margin distance is set to 5 for two datasets. The pre-trained embedding is set to ComplEx for all models. During the training procedure, HMNet uses Adam with the initial learning rate as 0.0001 to update parameters. The size of the hidden layer in the BiLSTM structure is set to $2d$, where d is the embedding dimension of datasets. The β is set to 0.5 for both datasets. All learning parameters are randomly initialized. For GMatching, we employ max/mean pooling (denoted as MaxP/MeanP) to obtain the prototype vector of the relation in reference set. Following FSRL [28], the maximum score between a query instance and K instances in the reference set is also considered as the final ranking score of this query instance (denoted as Max). For MetaR, we use pre-trained mode to maintain a consistent experimental environment. The results reported in the paper [28] are under the setting where the maximum size of the candidate set is 1000. Here, entities that satisfy type constraints [26] are added to the candidate set, where all candidates are considered in our work. In the absence of specific knowledge to choose otherwise, K is set to 5.

5.2 Results

The performance comparison results on two datasets are presented in Table 2, where the best results are shown in bold. We have following observations:

[3] https://github.com/DeepGraphLearning/KnowledgeGraphEmbedding.

Table 2. Link prediction results on two datasets. Results with * are reported in [26].

Model	NELL-One				Wiki-One			
	MRR	Hits@10	Hits@5	Hits@1	MRR	Hits@10	Hits@5	Hits@1
TransE [2]	.131	.220	.182	.081	.144	.204	.172	.108
RESCAL [15]	.033	.055	.038	.019	.060	.112	.081	.029
DistMult [27]	.051	.134	.081	.010	.027	.058	.035	.010
ComplEx* [20]	.200	.325	.269	.133	.033	.066	.046	.015
GMatching (MaxP) [26]	.189	.301	.225	.136	.134	.287	.181	.066
GMatching* (MeanP) [26]	.201	.311	.264	**.143**	.242	.419	.318	.163
GMatching (Max) [26]	.190	.305	.247	.123	.125	.251	.167	.065
MetaR [4]	.164	.320	.252	.083	.220	.347	.287	.158
FSRL [28]	.184	.341	.248	.105	.126	.242	.154	.068
HMNet	**.209**	**.364**	**.296**	.129	**.294**	**.423**	**.353**	**.230**

- HMNet yields the best performance under most evaluation metrics. Taking Hits@10 and MRR as examples, HMNet improves over the strongest baselines w.r.t. Hits@10 by 6.74% in NELL-One and w.r.t. MRR by 21.48% in Wiki-One, respectively. In particular, HMNet yields 41.1% higher performance w.r.t. Hits@1 than GMatching on Wiki-One. This verifies the significance of entity perspective evaluation. Moreover, compared with the fixed weights used in the other three few-shot learning methods, HMNet verifies the effectiveness of the feature attention mechanism.
- It can be seen that graph embedding methods work poorly on relations that have only a few triplets to train. It demonstrates the limitations of previous graph embedding methods for few-shot link prediction.

5.3 Further Analysis

Impact of Few-Shot Size: We conduct experiments to analyze the impact of few-shot size settings with $K \in \{3, 4, 5, 6\}$. The test results on NELL-One of different methods measured using Hits@k and MRR are shown in Fig. 4. HMNet outperforms other baseline methods on most evaluation metrics in different few-shot size settings, indicating its stability on few-shot link prediction. In addition, the performance of most methods does not improve with the few-shot size increasing in this experimental setting. The reason may be that unrepresentative instances are added to the reference set, which are far away from the prototype representation of the relation.

Impact of Embedding Methods: To observe the impact of different embedding methods for relation representation, we compare the performance between our model HMNet and the latest method FSRL [28]. Figure 5 shows the results of HMNet and FSRL on NELL-One dataset. We can see that FSRL obtains the best performance when using ComplEx as the embedding method. Compared with FSRL, HMNet achieves better performance in four different embedding method settings. It further indicates the superior performance of our model in terms of few-shot link prediction in KGs.

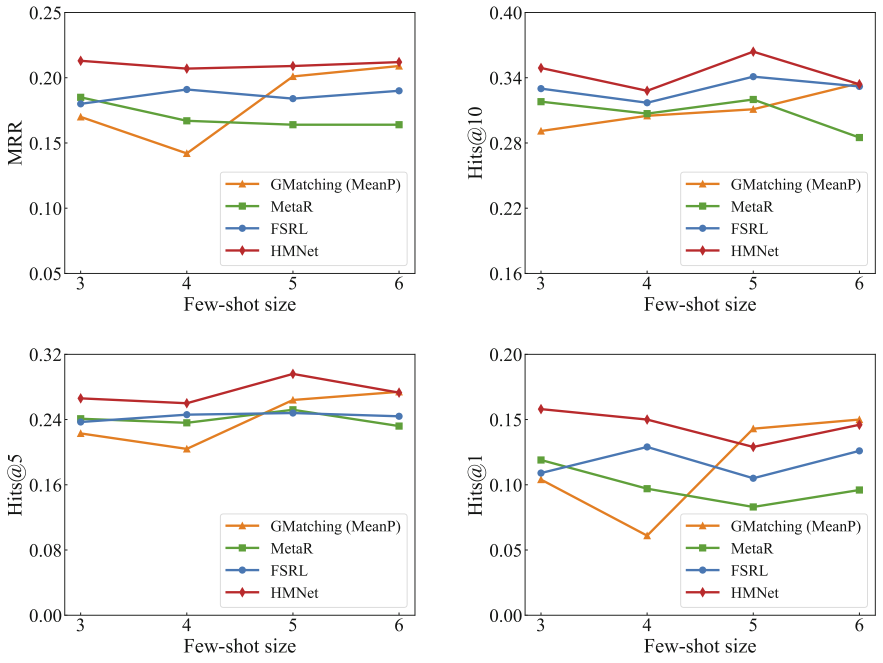


Fig. 4. Impact of few-shot size.

Table 3. Impact of hyperparameter β.

Hyperparameter β	MRR	Hits@10	Hits@5	Hits@1
0.2	.184	.325	.242	.112
0.5	**.209**	**.364**	**.296**	**.129**
1.0	.184	.356	.233	.111

On Parameter Selection for HMNet: We investigate the impact of different entity-aware score weights β on the few-shot link prediction performance. We conduct the experiment with hyperparameter $\beta \in \{0.2, 0.5, 1.0\}$ while other factors are fixed. The results on NELL-One are reported in Table 3 with the best results bold. HMNet reaches the best performance with $\beta = 0.5$. Moreover, the performance of our model first improves and then declines when β increases. The reason is that the entities matching can provide a valuable evaluation indicator. However, when β is larger than 0.5, it greatly reduces the influence of relation perspective evaluation which harms the hybrid matching performance.

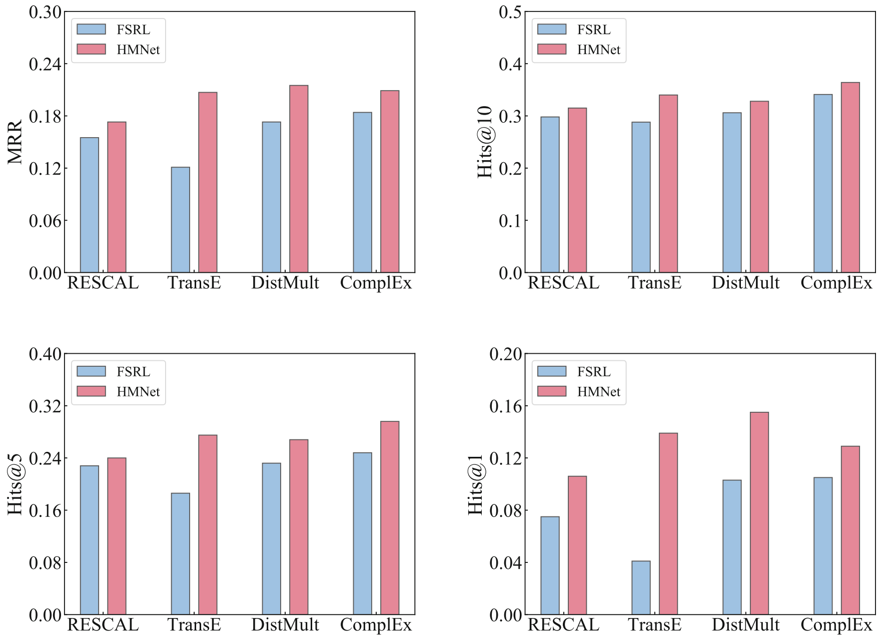


Fig. 5. Impact of embedding methods.

5.4 Ablation Study

HMNet consists of two components, and each component contains different modules. To get deep insight into HMNet, we analyze the contribution of each module. Specifically, we remove the entity-aware score module and only keep the AI Embed Layer (denoted as HMNetw/oEntityMatching). For the relation encoder module, we remove the BiLSTM network and only keep the multi-layer perceptron to get the representation of relation of each instance (denoted as HMNetw/oBiLSTM). To explore the impact of score function selection, we remove the feature attention module (denoted as HMNetw/oCNN). The parameters follow the above settings, and the results on NELL-One dataset are reported in Table 4 with the best results bold. Several observations from these results are worth noting:

- The best results of most evaluation metrics on the NELL-One dataset are obtained by complete HMNet.
- Removing the entity-aware score or CNN from the complete model causes the most significant performance drop on all evaluation metrics, showing the crucial role of entity perspective evaluation and the feature attention module in general.
- Removing the BiLSTM causes performance drop on some evaluation metrics but not all. All the components of HMNet together lead to the robust performance of our approach.

Table 4. Results of ablation study on NELL-One.

Model	MRR	Hits@10	Hits@5	Hits@1
HMNetw/oEntityMatching	.189	.326	.256	.119
HMNetw/oBiLSTM	.205	.352	.277	**.134**
HMNetw/oCNN	.193	.348	.263	.120
HMNet	**.209**	**.364**	**.296**	.129

6 Conclusion

In this paper, we propose a novel few-shot link prediction model, named HMNet. HMNet with entity-aware matching network and relation-aware matching network can evaluate the authenticity of triplets from two different perspectives. The comprehensive results on two public datasets indicate that HMNet can obtain more superior performance than state-of-the-art baseline methods. With in-depth analysis and ablation empirical evidence, we show the effectiveness and importance of each module of the HMNet model.

In the future, we will study the impact of different entity feature aggregation methods on experimental performance. Furthermore, we plan to integrate extra information (e.g., text information) to improve performance.

References

1. Bollacker, K.D., Evans, C., Paritosh, P., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: SIGMOD, pp. 1247–1250 (2008)
2. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS, pp. 2787–2795 (2013)
3. Bordes, A., Weston, J., Usunier, N.: Open question answering with weakly supervised embedding models. In: ECML-PKDD, pp. 165–180 (2014)
4. Chen, M., Zhang, W., Zhang, W., Chen, Q., Chen, H.: Meta relational learning for few-shot link prediction in knowledge graphs. In: EMNLP-IJCNLP, pp. 4216–4225 (2019)
5. Dalton, J., Dietz, L., Allan, J.: Entity query feature expansion using knowledge base links. In: SIGIR, pp. 365–374 (2014)
6. Dettmers, T., Minervini, P., Stenetorp, P., Riedel, S.: Convolutional 2D knowledge graph embeddings. In: AAAI, pp. 1811–1818 (2018)
7. Dong, X., et al.: Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In: SIGKDD, pp. 601–610 (2014)
8. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML, pp. 1126–1135 (2017)
9. Gao, T., Han, X., Liu, Z., Sun, M.: Hybrid attention-based prototypical networks for noisy few-shot relation classification. In: AAAI, pp. 6407–6414 (2019)
10. Hamilton, W.L., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. In: NIPS, pp. 1024–1034 (2017)

11. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
12. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: ICML (2015)
13. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI, pp. 2181–2187 (2015)
14. Munkhdalai, T., Yu, H.: Meta networks. In: ICML, pp. 2554–2563 (2017)
15. Nickel, M., Tresp, V., Kriegel, H.: A three-way model for collective learning on multi-relational data. In: ICML, pp. 809–816 (2011)
16. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
17. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: NIPS, pp. 4077–4087 (2017)
18. Suchanek, F.M., Kasneci, G., Weikum, G.: YAGO: a core of semantic knowledge. In: WWW, pp. 697–706 (2007)
19. Sun, Z., Deng, Z., Nie, J., Tang, J.: Rotate: Knowledge graph embedding by relational rotation in complex space. In: ICLR (2019)
20. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML, pp. 2071–2080 (2016)
21. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NIPS, pp. 3630–3638 (2016)
22. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.: KGAT: knowledge graph attention network for recommendation. In: KDD, pp. 950–958 (2019)
23. Wang, Y., Yao, Q., Kwok, J.T., Ni, L.M.: Generalizing from a few examples: a survey on few-shot learning. ACM Comput. Surv. **53**(3), 63:1–63:34 (2020)
24. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI, pp. 1112–1119 (2014)
25. Xie, Y., Xiong, Y., Zhu, Y.: SAST-GNN: a self-attention based spatio-temporal graph neural network for traffic prediction. In: DASFAA, pp. 707–714 (2020)
26. Xiong, W., Yu, M., Chang, S., Guo, X., Wang, W.Y.: One-shot relational learning for knowledge graphs. In: EMNLP, pp. 1980–1990 (2018)
27. Yang, B., Yih, W., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: ICLR (2015)
28. Zhang, C., Yao, H., Huang, C., Jiang, M., Li, Z., Chawla, N.V.: Few-shot knowledge graph completion. In: AAAI, pp. 3041–3048 (2020)
29. Zhang, F., Yuan, N.J., Lian, D., Xie, X., Ma, W.: Collaborative knowledge base embedding for recommender systems. In: SIGKDD, pp. 353–362 (2016)
30. Zhang, Z., Zhuang, F., Zhu, H., Shi, Z., Xiong, H., He, Q.: Relational graph neural network with hierarchical attention for knowledge graph completion. In: AAAI, pp. 9612–9619 (2020)