Adaptive Multi-hop Neighbor Selection for Few-Shot Knowledge Graph Completion

Xing Gong[1], Jianyang Qin[1], Ye Ding[2], Yan Jia[1,3], and Qing Liao[1,3](✉)

[1] School of Computer Science and Technology,
Harbin Institute of Technology, Shenzhen, China
{gongxing,22b351005}@stu.hit.edu.cn, {jiayan2020,liaoqing}@hit.edu.cn
[2] School of Cyberspace Security,
Dongguan University of Technology, Dongguan, China
dingye@dgut.edu.cn
[3] Peng Cheng Laboratory, Shenzhen, China

Abstract. Few-shot Knowledge Graph Completion (FKGC) is a special task proposed for the relations with only a few triples. However, existing FKGC models face the following two issues: 1) these models cannot fully exploit the dynamic relation and entity properties of neighbors to generate discriminative representations; 2) these models cannot filter out noise in high-order neighbors to obtain reliable entity representations. In this paper, we propose an **a**daptive **m**ulti-hop neigh**b**or se**l**ection model, namely AMBLE, to mitigate these two issues. Specifically, AMBLE first introduces a query-aware graph attention network (QAGAT) to obtain entity representations by dynamically aggregating one-hop neighbors based on relations and entities. Then, AMBLE aggregates high-order neighbors by iterating QAGAT and LSTM, which can efficiently extract useful and filter noisy information. Moreover, a Transformer encoder is used to learn the representations of subject and object entity pairs. Finally, we build an attentional matching network to map the query to few support triples. Experiments show that AMBLE outperforms state-of-the-art baselines on two public datasets.

Keywords: Knowledge graph completion · Few-shot learning · Link prediction · Graph attention network · Multi-hop aggregation

1 Introduction

Knowledge graphs (KGs) as a kind of structured data can assist many artificial intelligence downstream applications, such as question answering systems [23], recommendation systems [18], etc. KGs usually represent every fact with a triple (s, r, o), where s, o are the subject entity and object entity, and r is the relation between s and o. Due to the incompleteness of KGs, knowledge graph completion (KGC) has become one of the most important research tasks in the field of

B. Luo et al. (Eds.): ICONIP 2023, CCIS 1965, pp. 153–164, 2024.
https://doi.org/10.1007/978-981-99-8145-8_13

knowledge graphs. The task is to infer missing facts based on the existing entity and relation by answering queries such as $(s, r, ?)$.

Existing large-scale knowledge graphs [2,17] often suffer from the long-tail distribution problem, i.e., a large number of relations contain only a few triples. However, traditional KGC models require a large number of triples for each relation for training to obtain discriminative representations. As a result, these models have poor completion ability for the relation with only a small number of triples. To alleviate this problem, few-shot knowledge graph completion (FKGC) has been proposed recently. These methods only need a small number of triples as references for queries of each relation to achieve the completion task in few-shot scenarios.

Existing FKGC models [12,13,21,22] obtain entity representations by aggregating neighbor information, but these methods have two major limitations. **1) Dynamic neighbor properties:** Dynamic neighbor properties mean that the influence of neighbors on entity varies with the relation of different completion tasks. Dynamic neighbor properties are determined by both the entity and relation information. However, GMatching [21], FSRL [22] and GANA [12] ignore the dynamic properties of entities, these methods cannot dynamically assign neighbor weights based on the completion task, resulting in inaccurate encoding of entities. Although FAAN [13] considers the problem of dynamic neighbor properties, it only considers the effect of relations and ignores the effect of entities. **2) High-order neighbor noise:** In real-world knowledge graph datasets [2,17], there are a large number of entities that contain only a very small number of one-hop neighbors. Existing FKGC approaches only aggregate one-hop neighbors, resulting in their inability to obtain reliable representations of entities. Although traditional KGC model [11] aggregates high-order neighbors to obtain supplementary neighbor information, it ignores the noise problem in high-order neighbors. Thus, how to efficiently filter out these noisy high-order neighbors remains a challenging problem.

To address the above problems, we propose an **A**daptive **M**ulti-hop neigh**B**or se**LE**ction for few-shot knowledge graph completion (AMBLE). Specifically, we firstly propose a query-aware graph attention network (QAGAT) to obtain entity representations by aggregating one-hop neighbors. QAGAT can fully make use of both entity and relation information to dynamically assign weights to the neighbors. Secondly, we iterate QAGAT and LSTM to aggregate high-order neighbors which can effectively extract useful information from high-order neighbors and filter out the noisy information. Thirdly, we use a Transformer to learn the representations of entity pairs. Finally, an attentional matching network is applied to calculate the score of each query. Main contributions of this paper are summarized as follows:

- We propose an adaptive multi-hop neighbor selection model, namely AMBLE, to solve dynamic neighbor properties and high-order neighbor noise problems in FKGC.

- We devise a novel query-aware graph attention network (QAGAT), which can take advantage of both entity and relation information to adaptively assign neighbor weights based on different tasks.
- We design a new high-order neighbor aggregation and selection structure by iterating QAGAT and LSTM, which can efficiently extract useful and filter noisy information from high-order neighbors.
- We demonstrate the superiority of the AMBLE over state-of-the-art baselines by conducting extensive experiments on two public datasets.

2 Related Work

Due to the long-tail phenomenon in real-world KGs, FKGC has become a popular research area. Existing FKGC models can be grouped into two categories:

Metric-Based Models. GMatching [21] learns entity representations by aggregating neighbors, and then introduces a matching processor to evaluate the similarity between queries and support triples. FSRL [22] introduces a neighbor aggregator based on attention mechanism to aggregate neighbor information. FAAN [13] considers the problem of dynamic properties in neighbors and proposes a relation-aware attentional neighbor aggregator to learn entity representations. Thus, it can dynamically aggregate neighbors with the change of the completion task. YANA [8] aims to mitigate the issue of generating reliable embeddings for solitary entities in FKGC tasks, and introduces four novel abstract relations to represent inner- and cross- pair entity correlations and constructs a local pattern graph from the entities. MFEN [20] aims to capture the heterogeneous influence of neighbor characteristics by devising a single-layer CNN with differently sized filters to capture multi-scale characteristics while controlling model complexity.

Optimization-Based Models. MetaR [3] designs a fast gradient descent update procedure based on the idea of MAML [4] to achieve the completion task by transferring relational meta-information from support triples to queries. Based on MetaR, GANA [12] proposes a gated and attentive neighbor aggregator to filter noise in one-hop neighbors. In addition, benefiting from TransH [19], GANA designs a MTransH to deal with the complex relations.

However, the above models cannot utilize the information of entities and relations in dynamic properties problem on the one hand, and do not consider the noise problem in high-order neighbors on the other hand.

3 Preliminaries

In this section, we give formal definitions of the knowledge graph, the few-shot knowledge graph completion task, and the corresponding few-shot learning setting. Specific notations and their descriptions are listed in Table 1.

Table 1. Notations and descriptions.

Notation	Description
$\mathcal{G}, \mathcal{E}, \mathcal{R}, \mathcal{F}$	Knowledge Graph, entity set, relation set and fact set
(s, r, o)	A triple of subject entity, relation and object entity
$\mathcal{T}_i, \mathcal{S}_i, \mathcal{Q}_i$	Task i and its support set and query set
$\mathcal{Q}_i^+, \mathcal{Q}_i^-$	The positive and negative query set of task i
$\mathcal{G}^{'}, \mathcal{C}$	Background knowledge graph and candidate entity set
$\mathcal{N}_e$	Neighbor set of entity e
$\mathbf{e}^{(t)}, \mathbf{e}_i^{(t)}$	t-layer entity e and its neighbors' representations
$\mathbf{r}, \mathbf{r}_i$	Query relation and neighbor's relation representations
$\mathbf{q}, \mathbf{s}_i$	Query and support entity pairs representations
$\mathbf{W}_1, \mathbf{W}_2, \mathbf{W}_3$	$d \times d$ dimensional parameters of liner
b_1, b_2, b_3	d dimensional bias of liner
$\mathbf{W}_f^{(t)}, \mathbf{W}_i^{(t)}, \mathbf{W}_o^{(t)}, \mathbf{W}_C^{(t)}$	$d \times d$ dimensional parameters of t-layer LSTM
$b_f^{(t)}, b_i^{(t)}, b_o^{(t)}, b_C^{(t)}$	d dimensional bias of t-layer LSTM
σ	Activation function *sigmoid*
λ, γ	Hyperparameters

Knowledge Graph. A knowledge graph $\mathcal{G}$ is represented by a collection of triples: $\mathcal{G} = \{(s, r, o)|s, o \in \mathcal{E}, r \in \mathcal{R}\}$. For each triple (s, r, o), s, o denote the subject entity and object entity, and r is the relation between s and o. $\mathcal{E}$, $\mathcal{R}$ denote the entity set and relation set of $\mathcal{G}$, respectively.

Few-Shot Knowledge Graph Completion. Few-shot knowledge graph completion is a specialized task proposed for the relations with only a few triples, which are called few-shot relations. Each few-shot relation r corresponds to one knowledge graph completion task $\mathcal{T}_r$. Each task have a support set and a query set, i.e., $\mathcal{T}_r = \{\mathcal{S}_r, \mathcal{Q}_r\}$. Support set $\mathcal{S}_r = \{(s_i, r, o_i)|(s_i, r, o_i) \in \mathcal{G}\}$ contains support triples of task $\mathcal{T}_r$, and $|\mathcal{S}_r| = K$ suggests a K-shot knowledge graph completion task. Besides, query set $\mathcal{Q}_r$ contains all query triples of task $\mathcal{T}_r$, including positive query triples $\mathcal{Q}_r^+ = \{(s_i, r, o_i^+)|(s_i, r, o_i^+) \in \mathcal{G}, o_i^+ \in \mathcal{C}\}$ and corresponding negative query triples $\mathcal{Q}_r^- = \{(s_i, r, o_i^-)|(s_i, r, o_i^-) \notin \mathcal{G}, o_i^- \in \mathcal{C}\}$. $\mathcal{C}$ is the candidate entity set. A few-shot knowledge graph completion task is to find the best completion entity for each query from the candidate entity set using the support set as a reference.

Few-Shot Learning Setting. We follow the same few-shot settings proposed by GMatching [21]. We divide all few-shot relations into three disjoint subsets $\mathcal{R}_{train}$, $\mathcal{R}_{valid}$ and $\mathcal{R}_{test}$ for model training, validation and testing. Therefore, the training, validation and testing phases of our model correspond to a series of

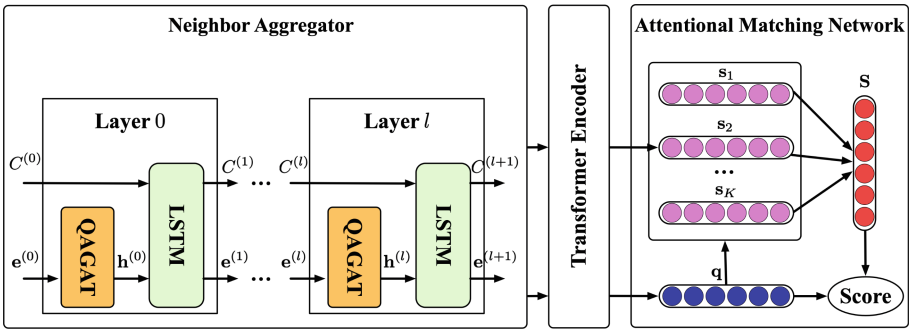

Fig. 1. Overall framework of our model AMBLE.

few-shot knowledge graph completion tasks. The training, validation and testing phases are defined as $\mathcal{T}_{train} = \{\mathcal{T}_i | i \in \mathcal{R}_{train}\}$, $\mathcal{T}_{valid} = \{\mathcal{T}_i | i \in \mathcal{R}_{valid}\}$ and $\mathcal{T}_{test} = \{\mathcal{T}_i | i \in \mathcal{R}_{test}\}$ respectively. In addition to few-shot relations, the other relations in knowledge graph $\mathcal{G}$ have sufficient triples, called high-frequency relations, and the triples containing high-frequency relations constitutes the background knowledge graph $\mathcal{G}'$.

4 Methodology

Given a task $\mathcal{T}_i = \{\mathcal{S}_i, \mathcal{Q}_i\}$, the purpose of AMBLE is to find the best candidate entity by matching the input query $q \in \mathcal{Q}_i$ to the given support set $\mathcal{S}_i$. To achieve this goal, as shown in Fig. 1, AMBLE consists of three major parts: (1) Neighbor aggregator to learn entity representations by aggregating neighbor information; (2) Transformer encoder to learn relational representations for entity pairs; (3) Attentional matching network to match the query to the given support set. Finally, we present the loss function and training details of our model.

4.1 Neighbor Aggregator

Neighbor aggregator is proposed to learn entity representations, which aggregate high-order neighbors through multiple layers of iterative aggregation. Each aggregation layer of the neighbor aggregator is shown in Fig. 2, and each layer consists of QAGAT and LSTM.

Query-Aware Graph Attention Network. The influence of neighbors on one entity keeps changing with the relation of current task, i.e., when completing different queries, neighbors have different weights of influence on the target entity. This dynamic property depends on the relevance between the target entity and neighbor entity on the one hand, and on the relevance between neighbor relation and query relation on the other hand. However, existing FKGC methods

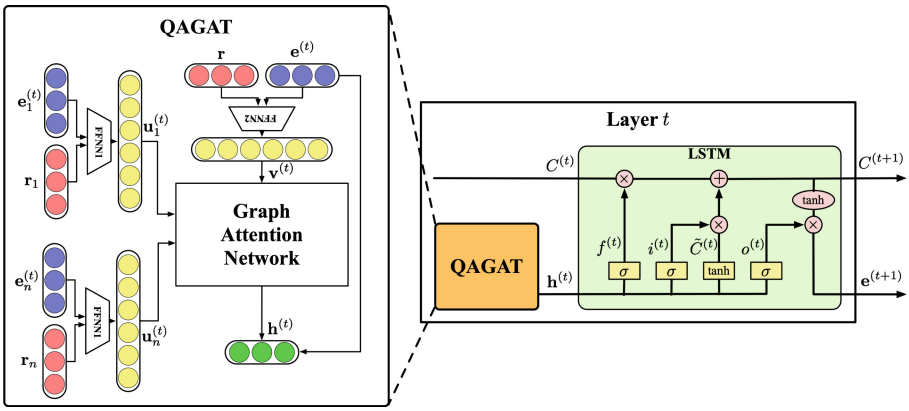

Fig. 2. Details of the t-layer of the neighbor aggregator

[12,13,21,22] cannot simultaneously consider these two influencing factors to obtain discriminative entity representations.

To tackle the above issue, we design a query-aware graph attention network (QAGAT) to dynamically aggregate one-hop neighbors. For each entity e, we constructs the neighbors of e, i.e., $\mathcal{N}_e = \{(e_i, r_i)|, (e, r_i, e_i) \in \mathcal{G}^{'}\}$, by searching for the triples in background knowledge graph $\mathcal{G}^{'}$ whose subject entity is e. e_i is the object entity considered as entity e's neighbor, r_i is the relation between e and e_i. At the t-layer, we first use two different FFNNs [14] to learn the integration representations of the neighbors and target entity as follows:

$$\mathbf{u}_i^{(t)} = \mathbf{W}_1(\mathbf{e}_i^{(t)} \| \mathbf{r}_i) + b_1 \tag{1}$$

$$\mathbf{v}^{(t)} = \mathbf{W}_2(\mathbf{e}^{(t)} \| \mathbf{r}) + b_2 \tag{2}$$

where $\|$ denotes concatenation operation. $\mathbf{u}_i^{(t)}$ denotes neighbor e_i's entity and relation integration representation, $\mathbf{v}^{(t)}$ denotes target entity e and query relation r integration representation. We obtain $\mathbf{r}$ given its support set $\mathcal{S}_r = \{(s_i, r, o_i)|s_i, o_i \in \mathcal{E}\}$ by TrasnE [1]: $\mathbf{r} = \mathbf{Mean}(\mathbf{e}_{o_i} - \mathbf{e}_{s_i})$.

Having obtained the integration representations of the neighbors and target entity, the weight $\alpha_i^{(t)}$ of the neighbor e_i for target entity e can be calculated as follows:

$$\alpha_i^{(t)} = \frac{\exp\ f(\mathbf{v}^{(t)}, \mathbf{u}_i^{(t)})}{\sum_{(e_j, r_j) \in \mathcal{N}_e} \exp\ f(\mathbf{v}^{(t)}, \mathbf{u}_j^{(t)})} \tag{3}$$

We use $softmax$ function to apply over $f(x, y)$, and we want to take benefit of the relevance of both entities and relations, so the $f(x, y)$ is defined as follows:

$$f(x, y) = LeakyReLU(x^\top \mathbf{W}_3 y + b_3) \tag{4}$$

$\mathbf{W}_3$ is the similarity matrix to calculate the relevance between x and y. Following GAT [15], we use activation function $LeakyReLU$ [10] here. Thus, we make full use of entity and relation information to dynamically assign neighbor weights.

Then, we can learn entity e's representation by adaptively aggregating neighbor information and its own information as follows:

$$\mathbf{h}^{(t)} = Relu(\lambda \sum_{(e_i,r_i)\in\mathcal{N}_e} \alpha_i^{(t)}\mathbf{e}_i^{(t)} + (1-\lambda)\mathbf{e}^{(t)}) \tag{5}$$

where $\mathbf{h}^{(t)}$ is the output representation of entity e at t-layer QAGAT. λ is a trade-off hyperparameter, and $Relu$ denotes activation function. Thus, QAGAT adaptively aggregates one-hop neighbors using both entity and relation information.

Adaptive Multi-hop Neighbor Selection. We expand from aggregating one-hop neighbors to aggregating multi-hop neighbors, aiming to find more complementary information from high-order neighbors. However, as the distance increases, more noise information is trapped in high-order neighbors [9]. To solve the high-order neighbor noise problem, we add a LSTM [6] after QAGAT of each layer aggregator for information filtering, because LSTM has excellent memory and forgetting functions. As shown in Fig. 2, the detailed calculation process for each step is as follows:

$$f^{(t)} = \sigma(\mathbf{W}_f^{(t)}\mathbf{h}^{(t)} + b_f^{(t)}),\ i^{(t)} = \sigma(\mathbf{W}_i^{(t)}\mathbf{h}^{(t)} + b_i^{(t)}),\ o^{(t)} = \sigma(\mathbf{W}_o^{(t)}\mathbf{h}^{(t)} + b_o^{(t)}) \tag{6}$$

$$\tilde{C}^{(t)} = tanh(\mathbf{W}_C^{(t)}\mathbf{h}^{(t)} + b_C^{(t)}),\ C^{(t+1)} = f^{(t)} \cdot C^{(t)} + i^{(t)} \cdot \tilde{C}^{(t)} \tag{7}$$

$$\mathbf{e}^{(t+1)} = o^{(t)} \cdot tanh(C^{(t+1)}) \tag{8}$$

where $\tilde{C}^{(t)}$ is the newly added neighbor information in the t-layer aggregation. The gated $i^{(t)}$ is used to extract useful information from newly added neighbors, and the extracted information is added to the memory by $i^{(t)} \cdot \tilde{C}^{(t)}$. The gated $f^{(t)}$ is used to filter noisy information from the old memory by $f^{(t)} \cdot C^{(t)}$. As such, we are able to filter the memory for entity e as $C^{(t+1)}$. The gated unit $o^{(t)}$ is used to select the output information from $C^{(t+1)}$. Then, we obtain representation $\mathbf{e}^{(t+1)}$ of the t-layer aggregation of entity e. After l-layer aggregation, we aggregate l-hop neighbor information and obtain entity e's final representation $\mathbf{e}^{(l)}$.

Through the above, we effectively extract useful information from the t-hop neighbors and filter out the noisy information. Therefore, the neighbor aggregator of our model can efficiently achieve the information aggregation and selection of high-order neighbors by iterating QAGAT and LSTM.

4.2 Transformer Encoder

With the neighbor aggregator, we have obtained the entity representation. Inspired by FAAN [13], which uses a Transformer module to learn the representation of entity pairs. We use a Transformer encoder to learn representations of entity pairs. We use the encoder to interact information between subject and object entities to learn more reliable representations of entity pairs. For each

triple (s, r, o) in support set or query set, we input them into Transformer as follows:

$$\mathbf{z}_1^0 = \mathbf{e}_s^{(l)} + \mathbf{x}_1^{pos}, \mathbf{z}_2^0 = \mathbf{r}_{mask} + \mathbf{x}_2^{pos}, \mathbf{z}_3^0 = \mathbf{e}_o^{(l)} + \mathbf{x}_3^{pos} \tag{9}$$

where $\mathbf{e}_s^{(l)}$ and $\mathbf{e}_o^{(l)}$ are the representations of entity s and o obtained by neighbor aggregator. $\mathbf{r}_{mask}$ is a randomly initialized mask. $\mathbf{x}_1^{pos}$, $\mathbf{x}_2^{pos}$ and $\mathbf{x}_3^{pos}$ are the position embeddings. Later, we feed them into a stack of P Transformer blocks as follows:

$$\mathbf{z}_i^p = Transformer(\mathbf{z}_i^{p-1}), i = 1, 2, 3. \tag{10}$$

where $\mathbf{z}_i^p$ is the hidden state of the p-th layer Transformer. After P layer Transformer, the final hidden state $\mathbf{z}_2^P$ is the representation of entity pair (s, o). By this way, we can obtain the representations $(\mathbf{s_1}, \mathbf{s_2}, ..., \mathbf{s_K})$ of entity pairs in support set $\mathcal{S}_r$ and the query entity pairs representation $\mathbf{q}$ of task $\mathcal{T}_r$.

4.3 Attentional Matching Network

Having obtained the representations of entity pairs in support set and query set by Transformer encoder, we adopt the idea of matching network [16] to calculate the similarity between query and support set to achieve FKGC task. Due to the semantic divergence in support set, different support triples have different weights for a query [5]. To enable our model to dynamically aggregate support triples when matching different queries, we adopt the attentional matching network in FAAN [13]. The similarity score of query $\mathbf{q}$ is calculated as follows:

$$\beta_i = \frac{\exp(\mathbf{q}^\top \mathbf{s}_i)}{\sum_{j=1}^K \exp(\mathbf{q}^\top \mathbf{s}_j)}, \ \mathbf{S} = \sum_{i=1}^K \beta_i \mathbf{s}_i \tag{11}$$

$$\text{Score}(\mathbf{q}, \mathcal{S}_r) = \mathbf{S}^\top \mathbf{q} \tag{12}$$

where β_i denotes the attention weight of support triple (s_i, r, o_i), and $\mathbf{S}$ is support set representation. Thus, we can obtain adaptive support set representation for different queries by Eq. 11. We take the inner product of the representations of query and support set as their similarity score.

4.4 Loss Function

Our model is trained on a training task set $\mathcal{T}_{train}$ with the goal of high similarity scores for positive queries and low similarity scores for negative queries. The objective function is a hinge loss defined as follows:

$$\mathcal{L} = \sum_r^{\mathcal{R}} \sum_{q^+ \in \mathcal{Q}_r^+, q^- \in \mathcal{Q}_r^-} \left[\gamma + Score(\mathbf{q}^-, \mathcal{S}_r) - Score(\mathbf{q}^+, \mathcal{S}_r)\right]_+ \tag{13}$$

where γ is a hyperparameter represents safety margin distance, and $[x]_+ = max(0, x)$ is the standard hinge loss.

Table 2. Statistics of the experimental datasets.

Dataset	#Relation	#Entity	#Triples	#Task-Train	#Task-Valid	#Task-Test
NELL-One	358	68545	181109	51	5	11
Wiki-One	822	4838244	5859240	133	16	34

5 Experiments

5.1 Datasets and Baselines

We conduct experiments on two public datasets NELL-One and Wiki-One. Following GMatching [21], we regard relations containing more than 50 but less than 500 triples as few-shot relations, and others as high-frequency relations. The task relation ratios for training/validation/testing on NELL-One and Wiki-One are 51/5/11 and 133/16/34, respectively. The statistics of the two datasets are shown in Table 2.

The existing FKGC models: including GMatching [21], MetaR [3], FSRL [22], FAAN [13], GANA [12], YANA [8] and MFEM [20]. To evaluate the performance of our model and the baselines for FKGC task, we utilize two traditional metrics MRR and Hits@1/5/10 on both datasets. The results of GMatching and FSRL are derived from the paper of FAAN, and the results of the other FKGC models are obtained from their corresponding original papers.

5.2 Implementation

In our model, all entities and relations representations are initialized randomly with dimension of 100 and 50 for NELL-One and Wiki-One. The few-shot size K is set to 5 for the following experiments. The two hyperparameters of this model, margin γ and trade-off λ, are set to 10 and 0.6 respectively. For Neighbor Aggregator, our model aggregates 2-hop neighbors on both datasets to achieve optimal performance, i.e., $l = 2$. The number of Transformer layers P is set to 3 and 4 for NELL-One and Wiki-One respectively, and the number of attention heads is set to 4 and 8 respectively. We implement all experiments with PyTorch and use Adam optimizer [7] to optimize model parameters with a learning rate of 0.0001.

5.3 Experimental Comparison with Baselines

We compare AMBLE with baselines on NELL-One and Wiki-One datasets to evaluate the effectiveness of AMBLE. The performances of all models are reported in Table 3, where the best results are highlighted in bold, and the best performance of baselines is underlined. AMBLE achieves general improvements compared to the baselines. To be concrete, 1) For NELL-One dataset, AMBLE achieves an improvement of 6.1/4.9/11.4/8.3% in MRR/Hits@1/5/10 compared to the best performing baseline GANA. These results illustrate that it

Table 3. Experimental results for all methods. The best results are marked in **bold**, and the best results of the baseline are underline.

Models (5-shot)	NELL-One				Wiki-One			
	MRR	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10
GMatching (MaxP)	0.176	0.113	0.233	0.294	0.263	0.197	0.337	0.387
GMatching (MeanP)	0.141	0.080	0.201	0.272	0.254	0.193	0.314	0.374
GMatching (Max)	0.147	0.090	0.197	0.244	0.245	0.185	0.295	0.372
MetaR (Pre-train)	0.209	0.141	0.280	0.355	0.323	0.270	0.385	0.418
MetaR (In-train)	0.261	0.168	0.350	0.437	0.221	0.178	0.264	0.302
FSRL	0.153	0.073	0.212	0.319	0.158	0.097	0.206	0.287
FAAN	0.279	0.200	0.364	0.428	0.341	0.281	0.395	0.463
GANA	0.344	0.246	0.437	0.517	0.351	0.299	0.407	0.446
YANA	0.294	0.230	0.364	0.421	0.380	0.327	0.442	0.523
MFEN	0.310	0.236	0.369	0.443	0.331	0.253	0.398	0.470
AMBLE (Ours)	**0.365**	**0.258**	**0.487**	**0.560**	**0.392**	**0.335**	**0.463**	**0.546**

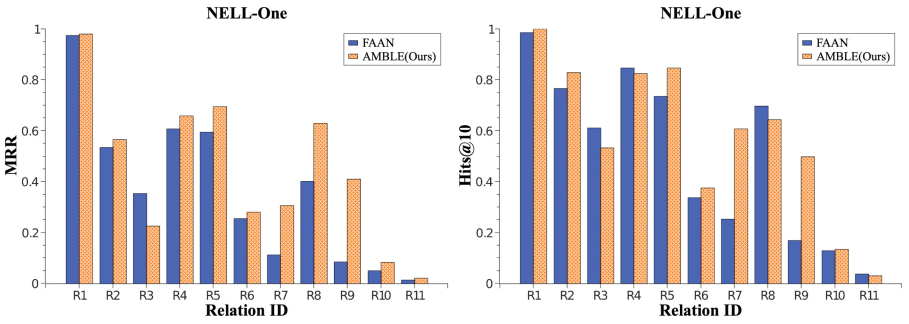

Fig. 3. The MMR and Hits@10 of AMBLE and FAAN for each relation on NELL-One.

is more effective to adaptively aggregate multi-hop neighbors based on relation and entity information. 2) For Wiki-One dataset, AMBLE achieves an improvement of 3.2/2.4/4.8/4.4% in MRR/Hits@1/5/10 compared to the best performing baseline YANA. Although YANA utilizes the information from the subgraphs, our model achieves a better performance. This indicates that our model can effectively extract useful and filter noisy information from high-order neighbors by iterating QAGAT and LSTM.

5.4 Comparison over Different Relations

To demonstrate the superiority of our model in more detail, we set up comparative experiments with FAAN [13] on NELL-One over different relations. The experimental results are shown in Fig. 3, where Relation ID represents a class of relation. AMBLE outperforms FAAN in MRR metric with 10 out of 11 relations,

Table 4. The results of ablation experiment.

Variants	NELL-One				Wiki-One			
	MRR	Hits@1	Hits@5	Hits@10	MRR	Hits@1	Hits@5	Hits@10
w/o QAGAT	0.265	0.187	0.334	0.408	0.342	0.254	0.388	0.463
w/o Neighbor selection	0.337	0.239	0.425	0.476	0.364	0.265	0.408	0.495
AMBLE (Ours)	**0.365**	**0.258**	**0.487**	**0.560**	**0.392**	**0.335**	**0.463**	**0.546**

and in Hits@10 metric with 7 out of 11 relations. Experimental results indicate that our model is robust for different task relations.

5.5 Ablation Study

We perform experiments on all the datasets with several variants of AMBLE to provide a better understanding of the contribution of each module to AMBLE. The ablative results are shown in Table 4. The experimental results demonstrate the effectiveness of each module in AMBLE. 1) **w/o QAGAT** means we replace QAGAT with the heterogeneous neighbor encoder in FSRL [22]. Experimental results show that QAGAT can obtain discriminative entity and relation representations by dynamically aggregating neighbors based on relation and entity information. 2) **w/o Neighbor selection** means that we remove the LSTM from the neighbor aggregator. Experimental results prove that using high-order neighbor information can improve the performance of our model. In addition, neighbor selection by LSTM can effectively extract useful and filter noisy information from high-order neighbors.

6 Conclusion

In this paper, we propose a novel model AMBLE to address the dynamic neighbor properties and high-order neighbor noise issues in few-shot knowledge graph completion. We propose a query-aware graph attention network (QAGAT) to dynamically aggregate neighbors based on relation and entity information, so as to capture the dynamic neighbor properties in completion task. In addition, we iterate QAGAT and LSTM to aggregate multi-hop neighbors, which can efficiently extract useful and filter noisy information from high-order neighbors. The experimental results on the datasets NELL-One and Wiki-One show the superiority of our model and the effectiveness of each component of our model.

Acknowledgements. This work was partially supported by the National Natural Science Foundation of China (Grant No. U19A2067, 61976051) and Major Key Project of PCL (Grant No. PCL2021A09, PCL2021A02, PCL2022A03).

References

1. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS (2013)
2. Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Hruschka, E., Mitchell, T.: Toward an architecture for never-ending language learning. In: AAAI, pp. 1306–1313 (2010)
3. 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. 4217–4226 (2019)
4. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: ICML, pp. 1126–1135 (2017)
5. Gong, X., Qin, J., Chai, H., Ding, Y., Jia, Y., Liao, Q.: Temporal-relational matching network for few-shot temporal knowledge graph completion. In: DASFAA, pp. 768–783 (2023)
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. In: Neural Computation, pp. 1735–1780 (1997)
7. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
8. Liang, Y., Zhao, S., Cheng, B., Yin, Y., Yang, H.: Tackling solitary entities for few-shot knowledge graph completion. In: KSEM, pp. 227–239 (2022)
9. Liu, Z., Chen, C., Li, L., Zhou, J., Li, X., Song, L., Qi, Y.: GeniePath: graph neural networks with adaptive receptive paths. In: AAAI, pp. 4424–4431 (2019)
10. Maas, A.L., Hannun, A.Y., Ng, A.Y., et al.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of the ICML, p. 3 (2013)
11. Nathani, D., Chauhan, J., Sharma, C., Kaul, M.: Learning attention-based embeddings for relation prediction in knowledge graphs. In: ACL, pp. 4710–4723 (2019)
12. Niu, G., et al.: Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In: SIGIR, pp. 213–222 (2021)
13. Sheng, J., et al.: Adaptive attentional network for few-shot knowledge graph completion. In: EMNLP, pp. 1681–1691 (2020)
14. Svozil, D., Kvasnicka, V., Pospichal, J.: Introduction to multi-layer feed-forward neural networks. In: Chemometrics and Intelligent Laboratory Systems, pp. 43–62 (1997)
15. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
16. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: NIPS (2016)
17. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**, 78–85 (2014)
18. Wang, X., Wang, D., Xu, C., He, X., Cao, Y., Chua, T.S.: Explainable reasoning over knowledge graphs for recommendation. In: AAAI, pp. 5329–5336 (2019)
19. Wang, Z., Zhang, J., Feng, J., Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In: AAAI (2014)
20. Wu, T., Ma, H., Wang, C., Qiao, S., Zhang, L., Yu, S.: Heterogeneous representation learning and matching for few-shot relation prediction. Pattern Recogn. **131**, 108830 (2022)
21. Xiong, W., Yu, M., Chang, S., Guo, X., Wang, W.Y.: One-shot relational learning for knowledge graphs. In: EMNLP, pp. 1980–1990 (2018)
22. Zhang, C., Yao, H., Huang, C., Jiang, M., Li, Z.J., Chawla, N.: Few-shot knowledge graph completion. In: AAAI, pp. 3041–3048 (2020)
23. Zhang, Y., Dai, H., Kozareva, Z., Smola, A.J., Song, L.: Variational reasoning for question answering with knowledge graph. In: AAAI, pp. 1–8 (2018)