Learning Inter-Entity Interaction for Few-Shot Knowledge Graph Completion

Yuling Li, Kui Yu*, Xiaoling Huang, Yuhong Zhang
Key Laboratory of Knowledge Engineering with Big Data, Ministry of Education, Hefei, China
School of Computer Science and Information Engineering, Hefei University of Technology
lyl95@mail.hfut.edu.cn, {yukui, zhangyh}@hfut.edu.cn, hxl@chzu.edu.cn

Abstract

Few-shot knowledge graph completion (FKGC) aims to infer unknown fact triples of a relation using its few-shot reference entity pairs. Recent FKGC studies focus on learning semantic representations of entity pairs by separately encoding the neighborhoods of head and tail entities. Such practice, however, ignores the inter-entity interaction, resulting in low-discrimination representations for entity pairs, especially when these entity pairs are associated with 1-to-N, N-to-1, and N-to-N relations. To address this issue, this paper proposes a novel FKGC model, named Cross-Interaction Attention Network (CIAN) to investigate the inter-entity interaction between head and tail entities. Specifically, we first explore the interactions within entities by computing the attention between the task relation and each entity neighbor, and then model the interactions between head and tail entities by letting an entity to attend to the neighborhood of its paired entity. In this way, CIAN can figure out the relevant semantics between head and tail entities, thereby generating more discriminative representations for entity pairs. Extensive experiments on two public datasets show that CIAN outperforms several state-of-the-art methods. The source code is available at https://github.com/cjlyl/FKGC-CIAN.

1 Introduction

Knowledge graphs (KGs) like YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2008), and Wikidata (Vrandečić and Krötzsch, 2014) have been successfully applied to various knowledge-driven applications, such as question answering (Saxena et al., 2020), semantic search (Xiong et al., 2017), and information retrieval (Liu et al., 2018). A knowledge graph consists of a large number of fact triples, where each triple is represented in the form of (head entity, relation, tail entity). Although typical KGs are large in size, they still suffer from incompleteness. This motivates the research in knowledge graph completion (KGC), which aims to predict the missing elements in incomplete triples.

In literature, state-of-the-art KGC models are usually based on knowledge graph embeddings (KGE). The key idea behind KGE-based methods is to embed entities and relations into low-dimensional vector spaces and measure the plausibility of triples based on their embeddings (Bordes et al., 2013; Nickel et al., 2011). Despite their success, existing KGE-based models generally require sufficient training triples for all relations. However, in many real-world KGs, a large proportion of (few-shot) relations only contain a limited number of triples (Xiong et al., 2018; Chen et al., 2019; Sheng et al., 2020). The shortage of training triples hinders existing KGE-based models from achieving satisfactory performance on these few-shot relations.

To address this issue, few-shot knowledge graph
Completion (FKGC) models have been proposed to predict a tail entity $t$ in a query triple $(h, r, ?)$ given $K$ support entity pairs about the task relation $r$. Their primary focus lies in learning semantic representations of entity pairs. Typically, most methods learn entity embeddings from their respective neighborhoods and then concatenate the embeddings of head and tail entities \citep{niu2021few, wang2021transform, zhang2020few}. Such practice, however, treats each entity independently when modeling the semantic information of entity pairs, ignoring the interactions between head and tail entities. As a result, the generated representations of entity pairs are not sufficiently discriminative, especially in 1-to-N, N-to-1, and N-to-N relations where many different entity pairs involve common entities.

To solve this problem, we propose to investigate the \textit{inter-entity interaction} between head and tail entities, which affects the importance of their semantic attributes to represent entity pairs. The basic idea behind this is that entity pairs are supposed to pay more attention to the relevant semantic attributes between head and tail entities. Figure 1 shows an example of the head entity GeorgeBush and its two entity pairs belonging to the task relation HasJobPosition. The left neighborhood (dashed blue) describes the political attribute of GeorgeBush as a former president of the US, while the right one (solid green) shows his soldier attribute as a navy pilot. Intuitively, the political attribute could be more helpful for representing (GeorgeBush, President), while the soldier attribute could contribute more to representing (GeorgeBush, Pilot). Hence, exploiting the interactions between head and tail entities is beneficial to characterize entity pairs.

To this end, we propose a \textbf{Cross-Interaction Attention Network} (CIAN) for few-shot knowledge graph completion. Our contributions can be summarized as follows:

- We investigate the \textit{inter-entity interaction} in few-shot knowledge graph completion, which differs from previous paradigms by exploring the relevant semantic information between head and tail entities to represent corresponding entity pairs.

- We model the inter-entity interaction through two stages: the interactions within entities are first captured to extract task-relevant entity attributes, and then the interactions between entities are explored to discern the attributes of one entity related to its paired entity.

- We conduct extensive experiments on the NELL-One and Wiki-One datasets. Experimental results demonstrate that our method outperforms the state-of-the-art methods of few-shot KG completion with different few-shot sizes.

2 \textbf{Related Work}

A common approach for KGC is to represent entities and relations as low-dimensional vectors (a.k.a., knowledge graph embeddings, KGE) and then learn a well-designed score function to measure the plausibility of triples in the embedding space. Existing KGE-based models can be roughly classified into two groups: (1) translational distance approaches, which view relations as translation operations and define a distance-based score function accordingly \citep{bordes2013translating, wang2014knowledge, lin2015learning}, and (2) semantic matching approaches, which establish similarity-based score functions and make predictions by matching latent semantics of entities and relations \citep{nickel2011three, trouillon2016complex, vashishth2019end, schlichtkrull2018modeling}. These KGE-based models usually assume that there are sufficient training triples for all relations, and thus are limited in practical few-shot scenarios.

Recently, much research progress has been made in few-shot KG completion, which solves KGC for unseen relations with a handful of training samples, based on some prior knowledge gained from a large number of similar but different relations. Existing FKGC approaches can be divided into two main categories: (1) Metric learning-based methods: GMatching \citep{xiong2018few} is the first study to solve the one-shot KG completion problem. It learns entity embeddings by encoding their direct neighbors and measures the similarities between the query and reference entity pairs by a multi-step matching processor. FSRL \citep{zhang2020few} extends GMatching to a few-shot scenario through a recurrent autoencoder aggregation network and employs a heterogeneous neighbor encoder to improve the quality of entity embeddings. Unlike GMatching and FSRL that learn the identical representation for the same entity in various triples, FAAN \citep{sheng2020few} allows for the dynamic properties of entities under different task
relations and learns task-aware entity representations. In addition, P-INT (Xu et al., 2021) leverages the paths from head to tail entities to represent entity pairs and calculates the path interactions between the support and query entity pairs. (2) Meta learner-based methods: MetaR (Chen et al., 2019) generates relation-specific meta-information using the embeddings of head and tail entities and updates relation meta via gradient meta. GANA (Niu et al., 2021) learns relation-meta by incorporating the neighborhood information of entities and devises a meta-learning-based TransH module to model complex relations.

Different from the aforementioned methods that treat each entity independently, our work explores the inter-entity interaction and leverages the relevant attributes between entities to represent entity pairs.

3 Problem Formulation

A KG can be represented as a set of triples $G = \{(h, r, t) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}\}$, where $\mathcal{E}$ and $\mathcal{R}$ are the entity set and the relation set, respectively. Given any two of three elements within a triple, the KG completion task aims to predict the remaining one. This study focuses on predicting $t$ given $(h, r, ?)$ since our goal is to infer new facts for few-shot relations.

Following the standard setting in FKGC (Xiong et al., 2018; Chen et al., 2019), we have access to a background KG $G'$, which is a subset of $G$ with some relations of $\mathcal{R}$. The remaining relations in $\mathcal{R}$ are further divided into three disjoint task sets $\mathcal{R}_{train}$, $\mathcal{R}_{test}$ and $\mathcal{R}_{valid}$, which are used in the meta-training, meta-testing, and meta-validation phases, respectively.

To perform few-shot learning, we adopt the episodic paradigm (Vinyals et al., 2016) to imitate the real test conditions. Specifically, in each iteration of the meta-training phase, we sample a task relation $r$ from $\mathcal{R}_{train}$, and construct its support set $S_r = \{(h_i, t_i)\}_{i=1}^K$ and its query set $Q_r = \{(h_j, t_j, C_{h_j, r})\}_{j=1}^B$. Here, $K$ and $B$ denote the numbers of support and query triples, $t_j$ is the ground-truth tail entity for query head entity $h$ about relation $r$, and $C_{h_j, r}$ is the set of the corresponding candidate entities in $G$. The candidate entities are constructed based on the entity type constraint (Xiong et al., 2018; Toutanova et al., 2015). Our goal is to train a model using $K$ support entity pairs in $S_r$, such that the model can rank the ground-truth entity $t_j$ higher than the corresponding candidate entities in $C_{h_j, r}$. A few-shot KG completion task can be defined as follows:

**Definition of Few-shot KG Completion.** *Given a task relation $r$ and its support set $S_r = \{(h_i, r, t_i)\}_{i=1}^K$, one task is to predict the missing tail entities of the query triples, where few-shot size $K$ is very small. The task is called K-shot KG completion.*

After sufficient training with $\mathcal{R}_{train}$, the learned model can be used for the meta-validation and meta-testing phases with $\mathcal{R}_{valid}$ and $\mathcal{R}_{test}$, which are defined in the same way as $\mathcal{R}_{train}$.

4 Methodology

The framework of the proposed CIAN is illustrated in Figure 2. CIAN includes the following stages: (1) modeling the inter-entity interaction by a task-aware attention module and an entity-pair-aware attention module to generate the semantic representations of entity pairs, and (2) matching support and query sets to compare the degree of semantic matching between the input queries and the given support entity pairs.

4.1 Modeling the Inter-entity Interaction

In this subsection, we aim to explore the inter-entity interaction between the head and tail entities and utilize their relevant semantics to represent the corresponding entity pairs. To achieve this goal, we take the following two steps: (1) First, we design a task-aware attention module to extract task-relevant semantic attributes for each entity. This step is motivated by FAAN (Sheng et al., 2020), which points out that entities should exhibit the semantic attributes that adapt to the task relations. (2) On the basis of the task-relevant attributes, we then explore the relevant semantic attributes between head and tail entities. This is realized by an entity-pair-aware attention module, which attempts to compute the attention of interaction between one entity and the neighbors of its paired entity. Figure 3 gives an example of modeling the inter-entity interaction. We detail each module in the following.

4.1.1 Task-aware Attention Module

Given a pair of entities $(h, t)$ about the task relation $r$, the task-aware attention module takes as input each entity and its local neighborhood, and outputs its task-aware representation.

---

7693
Take the head entity \( h \) as a target, and we denote its local neighborhood as \( \mathcal{N}_h = \{(r_{hi}, e_{hi}) | (h, r_{hi}, e_{hi}) \in G'\} \). We first leverage three linear transformations, parameterized by \( W^Q, W^K, W^V \in \mathbb{R}^{d \times d} \), to project the task relation representation into the query, the neighboring relation representations into the keys, and the semantic representations of neighbors into the values, as follows:

\[
\begin{align*}
Q^s &= r_{ht} W^Q, \\
K^s &= r_{hi} W^K, \\
V^s &= n_{hi} W^V, \\
\end{align*}
\]

(1)

where

\[
[r_{ht} || r_{th}] = \text{Bilinear}(h, t), \\
n_{hi} = \text{ReLU}([r_{ht} || e_{hi}] W_n + b_n).
\]

(2)

(3)

Here, \( h, t, e_{hi}, r_{hi} \in \mathbb{R}^d \) are the embedding vectors of \( h, t, e_{hi}, r_{hi} \), respectively; \( d \) is the embedding dimension; \( || \) represents the concatenation operation; \( W_n \in \mathbb{R}^{2d \times d} \) and \( b_n \in \mathbb{R}^d \) are learnable parameters shared by all neighbors. In this paper, we adopt a bilinear function to model pairwise interactions between the head and tail entities (as shown in Eq. 2), and obtain the forward representation \( r_{ht} \in \mathbb{R}^d \) and the backward representation \( r_{th} \in \mathbb{R}^d \) about the task relation \( r \). In addition, we apply a feed-forward neural network to learn the semantic representation \( n_{hi} \) of neighbor \( (r_{hi}, e_{hi}) \).

Different from previous studies (Sheng et al., 2020; Zhang et al., 2020) that only use the information of neighboring entities to represent neighbor semantics, we compute the semantic representations of neighbors by combining the information of neighboring entities and neighboring relations, as shown in Eq. 3.

Next, we calculate the relevance between the neighboring relation and the task relation by a bilinear dot product, and then normalize the relevance scores across all neighbors using the softmax function, as follows:

\[
\text{Att}^s_{hi} = Q^s W^s K^s T, \\
\alpha_{hi} = \frac{\exp(\text{Att}^s_{hi})}{\sum_{(r_{hj}, e_{hj}) \in \mathcal{N}_h} \exp(\text{Att}^s_{hj})},
\]

(4)

(5)

where \( W_s \in \mathbb{R}^{d \times d} \) is a trainable weight vector that captures interactions between relations; \( \alpha_{hi} \) represents the task-relevant attention weight of neighbor \( (r_{hi}, e_{hi}) \). Afterward, we obtain the task-aware neighbor representation \( \bar{h}_s \) about \( h \) by considering its diverse roles:

\[
\bar{h}_s = \sum_{(r_{hi}, e_{hi}) \in \mathcal{N}_h} \alpha_{hi} V^s_{hi}.
\]

(6)
In this way, entity neighbors that are more relevant to the task relation will be assigned higher weights, thus capturing more valuable neighbor information.

Finally, we incorporate the weighted neighbor representation $\hat{h}_s$ into the entity embedding $h$ to generate the task-aware entity representation:

$$h_s = \text{ReLU}(hW_{ent} + \hat{h}_sW_{nbr}),$$  \hspace{1cm} (7)

where $W_{ent}, W_{nbr} \in \mathbb{R}^{d \times d}$ are two trainable weight vectors.

Likewise, we also perform the above procedures on tail entity $t$ and its local neighborhood $N_t$, and obtain its task-aware entity representation $t_s$.

4.1.2 Entity-pair-aware Attention Module

Given a pair of entities, the entity-pair-aware attention module aims to identify the relevant semantics between head and tail entities by letting one entity attend to each neighbor of its paired entity. The generated entity-pair-aware representations are then coupled together to form the semantic representation of the corresponding entity pair.

Take the head entity $h$ as a target. The entity-pair-aware attention module takes as input $\{h, N_h, t_s\}$ and outputs the entity-pair-aware representation of $h$. In specific, for each neighbor $(r_{hi}, e_{hi}) \in N_h$, we transform the task-aware representation of $t$ into the query, the neighboring entity representation $e_{hi}$ into the keys, and the neighbor representation $n_{hi}$ into the values with three linear transformations, $W^Q, W^K, W^V \in \mathbb{R}^{d \times d}$, respectively, as follows:

$$Q^c_{hi} = t_sW^Q e_{hi},$$
$$K^c_{hi} = e_{hi}W^K e_{hi},$$
$$V^c_{hi} = n_{hi}W^V e_{hi}.$$  \hspace{1cm} (8)

Then, we compute the relevance of each neighbor with respect to $t$:

$$\text{Att}_{ch_{hi}} = Q^c_{hi}W_cK^c_{hi}^T,$$  \hspace{1cm} (9)

where $W_c \in \mathbb{R}^{d \times d}$ is a trainable weight vector that captures the interactions between entities. Afterward, we compute the entity-pair-aware neighbor representation $\hat{h}_c$ for $h$:

$$\hat{h}_c = \sum_{(r_{hi}, e_{hi}) \in N_h} \beta_{hi}V^c_{hi},$$

$$\beta_{hi} = \frac{\exp(\text{Att}_{ch_{hi}})}{\sum_{(r_{hj}, e_{hj}) \in N_h} \exp(\text{Att}_{ch_{hj}})},$$  \hspace{1cm} (10)

where $\beta_{hi}$ denotes the entity-pair-aware attention weight of neighbor $(r_{hi}, e_{hi})$. Higher values of $\beta_{hi}$ are indicative of stronger relevance between the $i$-th entity neighbors of $h$ and the paired entity $t$.

Finally, we generate the entity-pair-aware representation of $h$ by combining its entity embedding $h$ and its neighbor representation:

$$h_c = \text{ReLU}(hW_{ent} + \hat{h}_cW_{nbr}).$$  \hspace{1cm} (11)

Analogously, for tail entity $t$, the attention mechanism takes as input $\{t, N_t, h_s\}$ and generates its entity-pair-aware representation $t_c$. Based on the generated representations $h_c$ and $t_c$, we compute the semantic representation $p(h, t)$ of entity pair $(h, t)$ by a position-wise forward feed neural network:

$$p(h, t) = \text{ReLU}((h_c | t_c)W_{p1} + b_p)W_{p2},$$  \hspace{1cm} (12)

where $W_{p1}, W_{p2} \in \mathbb{R}^{2d \times 2d}$ and $b_p \in \mathbb{R}^{2d}$ are trainable matrix vectors.

4.2 Matching Support and Query Sets

Based on the generated semantic representations of entity pairs, we are going to learn a general representation from the representations of the reference entity pairs in the support set, i.e., the prototype of the few-shot relation. Due to the scarcity of reference triples, one reference triple far from other references in the vector space will cause a huge deviation of the corresponding prototype (Gao et al., 2019). Inspired by (Sheng et al., 2020; Gao et al., 2019), we introduce an attention mechanism to select more informative references and denoise the noisy references during training.

In specific, given a query $q_r$ in the query set $Q_r$, the prototype vector $Z_{q_r}$ of few-shot relation $r$ can be obtained as follows:

$$Z_{q_r} = \sum_{s_{rk} \in S_r} \gamma_k p_{s_{rk}},$$  \hspace{1cm} (13)

$$\gamma_k = \frac{\exp(p_{q_r \odot p_{s_{rk}}})}{\sum_{s_{ri} \in S_r} \exp(p_{q_r \odot p_{s_{ri}}})}.$$  \hspace{1cm} (14)

Here, $s_{rk} = (l_k, r, l_k) \in S_r$ denotes the $k$-th triple in the support set $S_r$; $\odot$ denotes element-wise production. As such, the prototype $Z_{q_r}$ focuses more on the query-relevant references and thus reduces the impact of noisy references.
<table>
<thead>
<tr>
<th>Dataset</th>
<th># Ent</th>
<th># Triple</th>
<th># Rel</th>
<th># Task</th>
</tr>
</thead>
<tbody>
<tr>
<td>NELL-One</td>
<td>68,545</td>
<td>181,109</td>
<td>358</td>
<td>67</td>
</tr>
<tr>
<td>Wiki-One</td>
<td>4,838,244</td>
<td>5,859,240</td>
<td>822</td>
<td>183</td>
</tr>
</tbody>
</table>

Table 1: Statistics of datasets. # Ent, # Triple, # Rel and # Task represent the number of entities, triples, relations, and task relations, respectively.

To predict the plausibility of \( q_r \), we calculate the similarity score between the query representation of \( q_r \) and the prototype vector of \( r \), as follows:

\[
\phi(q_r, S_r) = p_{q_r} \odot z_{q_r}.
\]

(15)

Higher values of \( \phi(\cdot) \) are indicative of more reasonable triples.

4.3 Loss Function and Model Training

Given a task \( r \in R_{train} \) and its corresponding triples, we randomly sample \( K \) entity pairs to construct the support set \( S_r \), and a batch of entity pairs to construct the positive query set \( Q_r \). For each entity pair in \( Q_r \), we pollute its tail entity and construct a set of negative query entity pairs \( Q^r = \{(h_q, t_q^r) | (h_q, t_q) \in Q_r\} \), where \( t_q^r \in \{C_{h_q,r} \setminus t_q\} \). Then we use a margin-based scoring function to ensure that a positive query in \( Q_r \) has a higher similarity score than a negative query in \( Q^r \), as follows:

\[
L = \sum_r \sum_{q_r \in Q_r} \sum_{q^r \in Q^r} [\gamma + \phi(q^{-}_r, S_r) - \phi(q_r, S_r)]_+,
\]

(16)

where \([x]_+ = \max(0, x)\) is the hinge loss, and \( \gamma > 0 \) is a margin hyperparameter. We adopt a batch sampling-based meta-training procedure (Vinyals et al., 2016) to minimize \( L \) and optimize model parameters.

5 Experiments

In this section, we conduct extensive experiments to evaluate the performance of the proposed CIAN and verify the effectiveness of the key components in CIAN.

5.1 Experimental Settings

Datasets. In our experiments, we choose two widely used benchmark datasets, namely NELL-One and Wiki-One (Xiong et al., 2018), for few-shot KG completion. In both datasets, relations with less than 500 but more than 50 triples are selected as few-shot relations, and the remaining relations along with their triples constitute the background knowledge graphs. With this setting, there are 67 and 183 few-shot relations on NELL-One and Wiki-One, respectively. Following the original settings (Xiong et al., 2018), we split the training/test/validation relations as 51/11/5 and 133/34/16 on NELL-One and Wiki-One, respectively. The detailed statistics of the two datasets are summarized in Table 1.

Evaluation Metrics. We evaluate the model performance using two common metrics: MRR and Hits@\( N \). MRR is the average of the reciprocal ranks of the correct entities. Hits@\( N \) is the proportion of correct entities ranked in the top \( N \), with \( N = 1, 5, 10 \). Higher values of MRR or Hits@\( N \) are indicative of the better performance of KG completion.

Comparison Methods. We choose the following 5 state-of-the-art models\(^1\) as the baselines to compare with CIAN: GMatching (Xiong et al., 2018), MetaR (Chen et al., 2019), FSRL (Zhang et al., 2020), FAAN (Sheng et al., 2020), and GANA (Niu et al., 2021) (cf. Section 2 for model details). All the above methods learn entity embeddings by independently encoding their neighborhoods. Traditional KGE-based models such as TransE, DisMult, and ComplEx have been shown to perform significantly worse than the aforementioned few-shot baselines (Sheng et al., 2020; Niu et al., 2021). Due to space limitations, we omit the comparisons with these traditional methods.

Implementation Details. We initialize the entity and relation embeddings with the pre-trained TransE embeddings provided by Xiong et al. (2018), and further fine-tune these embeddings during training. On both datasets, we set the maximum number of neighbors as 100, and fix the margin \( \gamma \) to 5.0. We use the Adam optimizer (Kingma and Ba, 2015) as the optimizer with an initial learning rate of \( 8e^{-5} \) for NELL-One and \( 2e^{-4} \) for Wiki-One, respectively. Following (Sheng et al., 2020), we evaluate our model on the validation set at every 10k training steps, and save the best model when MRR reaches the highest value within 300k steps. We select the optimal hyperparameters of our method by grid search on the validation set.

\(^1\)For the sake of fairness, we do not compare our method with P-INT (Xu et al., 2021) since its results are obtained by discarding disconnected entity pairs.
<table>
<thead>
<tr>
<th>Method</th>
<th>NELL-One</th>
<th>Wiki-One</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MRR</td>
<td>Hits@10</td>
</tr>
<tr>
<td></td>
<td>3-shot</td>
<td>5-shot</td>
</tr>
<tr>
<td>GMatching (Xiong et al., 2018)</td>
<td>-</td>
<td>.176‡</td>
</tr>
<tr>
<td>MetaR (In-Train) (Chen et al., 2019)</td>
<td>.245†</td>
<td>.261</td>
</tr>
<tr>
<td>MetaR (Pre-Train) (Chen et al., 2019)</td>
<td>.210†</td>
<td>.209</td>
</tr>
<tr>
<td>FSRL (Zhang et al., 2020)</td>
<td>.219†</td>
<td>.195</td>
</tr>
<tr>
<td>FAAN (Sheng et al., 2020)</td>
<td>.247†</td>
<td>.279</td>
</tr>
<tr>
<td>GANA (Niu et al., 2021)</td>
<td>.322</td>
<td>.344</td>
</tr>
<tr>
<td>CIAN</td>
<td>.344</td>
<td>.376</td>
</tr>
</tbody>
</table>

Table 2: Evaluation results of all methods on NELL-One and Wiki-One in terms of MRR and Hits@{1, 5, 10}.

5.2 Overall Comparison with Baselines

Table 2 reports the performance of all models on the NELL-One and Wiki-One datasets. From the table, we can observe that our CIAN significantly outperforms all baselines on both datasets. Specifically, for 5-shot link prediction, CIAN achieves relative performance improvements of 3.2% / 1.0% / 1.6% / 5.2% in MRR / Hits@10 / Hits@5 / Hits@1 on NELL-One, and relative improvements of 3.2% / 4.2% / 4.6% / 1.9% on Wiki-One, compared to the best performing baseline. For 3-shot link prediction, CIAN achieves relative improvements of 2.7% / 5.7% / 4.9% / 0.1% in MRR / Hits@10 / Hits@5 / Hits@1 on Wiki-One over the best performing baseline. These results demonstrate that exploiting the interactions between head and tail entities can indeed improve the performance of few-shot KG completion.

To further investigate the effectiveness of our model under different few-shot sizes, we compare the performance of all models on NELL-One in various settings of $K$. The results are shown in Figure 4. It can be observed that:

(1) CIAN consistently outperforms all baselines by a large margin under different $K$, further proving the superiority of our model in the few-shot scenario.

(2) With the increment of $K$, the performance of all models does not continuously increase, suggesting that larger reference sets do not always lead to better performance of FKGC models. Even so, CIAN still gets relatively stable improvements compared to most baselines such as FSRL and MetaR (In-Train).

5.3 Comparison with Variants

To evaluate the contributions of each component in the proposed CIAN, we compare CIAN with its three variant models. The results are displayed in Table 3. We observe that the performance drops significantly when the task-aware attention module is removed (-TAM). This demonstrates that suppressing task-irrelevant entity attributes can effectively
Table 3: Results of CIAN and its variants in 5-shot link prediction on NELL-One. “-TAM” and “-EPAM” represent CIAN without task-aware attention module, and without entity-pair-aware attention module, respectively. “-Eq.2” means that we replace Eq.2 with the equation \( r = t - h \) used in FAAN.

Table 4: Results of CIAN, FAAN and GANA for each relation (RID) in NELL testing data.

Figure 5: Embedding visualization of entity pairs about relation ProducedBy.

6 Conclusion

In this paper, we investigate the inter-entity interaction and propose a cross-interaction attention network for few-shot KG completion. Specifically, a task-aware attention module is designed to extract task-relevant entity attributes, and an entity-pair-aware attention module is developed to identify the relevant semantic attributes between head and tail entities. Experiments on two public datasets demonstrate that our model outperforms state-of-the-art models with different few-shot sizes. In the future, we will study how to make use of the side information such as textual descriptions of entities to solve the few-shot KG completion task.

Acknowledgements

This work is supported by the National Key Research and Development Program of China (under grant 2020AAA0106100), National Natural Science Foundation of China (under Grant 61876206), Open Project Foundation of Intelligent Informa-
tion Processing Key Laboratory of Shanxi Province (under grant CICIP2020003), and the Key Project of the Natural Science Foundation of Educational Commission of Anhui Province (under grant KJ2021A1065).

Limitations
Our method has the following limitations:

• The performance of our model may degrade when there are many long-tailed entities with few direct neighbors. This is because sparse neighborhoods are not beneficial to model the inter-entity interaction.

• Since our method focuses on exploiting the interactions between entity neighborhoods, it requires a background graph to provide the neighbors of entities.

References


