

# Exploring Entity Interactions for Few-Shot Relation Learning (Student Abstract)

Yi Liang<sup>1</sup>, Shuai Zhao<sup>1</sup>, Bo Cheng<sup>1</sup>, Yuwei Yin<sup>2</sup>, Hao Yang<sup>3</sup>

<sup>1</sup> State Key Laboratory of networking and switching technology,  
Beijing University of Posts and Telecommunications, Beijing, China

<sup>2</sup> 2012 Labs, Huawei Technologies, CO., LTD, Shenzhen, China

<sup>3</sup> 2012 Labs, Huawei Technologies, CO., LTD, Beijing, China

{liangyi, zhaoshuai, chengbo}@bupt.edu.cn,  
{yinyuwei, yanghao30}@huawei.com

## Abstract

Few-shot relation learning refers to infer facts for relations with a limited number of observed triples. Existing metric-learning methods for this problem mostly neglect entity interactions within and between triples. In this paper, we explore this kind of fine-grained semantic meanings and propose our model TransAM. Specifically, we serialize reference entities and query entities into sequence and apply transformer structure with local-global attention to capture both intra- and inter-triple entity interactions. Experiments on two public benchmark datasets NELL-One and Wiki-One with 1-shot setting prove the effectiveness of TransAM.

## Introduction

Knowledge graphs (KGs) store facts in the form of triples  $(h, r, t)$  reflecting the relation  $r$  between head  $h$  and tail entity  $t$ . Few-shot relation learning or few-shot knowledge graph completion (FSKGC), which aims to complete the relations with only a limited number of observed triples (few-shot references), has attracted research attention in recent years. However, existing FSKGC models mostly fail to model entity level interactions within and between triples as they perform matching over entity pair representations.

In this paper, we argue that entity interactions can provide valuable fine-grained semantic meanings and propose TransAM, Transformer(Vaswani et al. 2017) Appending Matcher with local-global attention for FSKGC. In particular, we serialize references of few-shot relation  $r$  and query pair following with special token [CLS] into entity sequence and feed to transformer. To capture intra-triple level entity interactions, we construct a block attention mask matrix to constrain that every entity only attends to triple it involves (local attention). Besides, we introduce rotary operation to encode head or tail roles for each entity to model relation patterns, *i.e.*, symmetry/anti-symmetry or inversion. Global attention focuses more on modelling inter-triple entity interactions. We devise a separated triple position encoding for the self-attention module to preserve the triple structure and decouple entity representations and triple position embedding as we concern they are two kinds of heterogeneous features. Finally, the representation of [CLS] is used

to predict the plausibility of entity sequence, which indicates whether query fact  $(h_q, r, t_q)$  holds.

## Proposed Method

**Task Definition** Knowledge graph  $\mathcal{G}$  is formulated as  $\mathcal{G} := \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$ .  $\mathcal{E}$  is entity set,  $\mathcal{R}$  is a set of relations and  $\mathcal{T}$  is the triple set. Background graph  $\mathcal{BG}$  is a set of known triples. Giving a new relation  $r$  with  $K$  entity pairs as support set  $\mathcal{S}_r = \{(h_i, t_i) | (h_i, r, t_i) \in \mathcal{G}\}$  and a query entity pair  $(h_q, t_q)$ , the goal of FSKGC is to estimate whether  $(h_q, r, t_q)$  is a missing fact. *In this work, we solve this problem by serializing support set and query pair to an entity sequence  $s_q := [\text{CLS}], h_1, t_1, \dots, h_K, t_K, h_q, t_q]$  and train the model to compute the probability of whether  $s_q$  holds.*

**Entity Encoder** Following previous work(Zhang et al. 2020), we enhance entity representations with a heterogeneous graph encoder. For each entity  $e$  from input sequence  $s_q$ , after extracting neighbors  $\mathcal{N}_e = \{(r_i, t_i)\}$  starting with  $e$  from  $\mathcal{BG}$ , we calculate the contribution  $\alpha_i = \frac{\exp(\mathbf{u}^T (\text{ReLU}(\mathbf{W}_1(\mathbf{v}_{r_i} \parallel \mathbf{v}_{t_i}))) + b_1)}{\sum_{j=1}^{|\mathcal{N}_e|} \exp(\mathbf{u}^T (\text{ReLU}(\mathbf{W}_1(\mathbf{v}_{r_j} \parallel \mathbf{v}_{t_j}))) + b_1)}$ . Then we aggregate neighbors information following  $\mathbf{h}_e = \sum_{i=1}^{|\mathcal{N}_e|} \alpha_i \mathbf{v}_{t_i}$ . Further, a fusion mechanism is applied to preserve origin embedding  $\mathbf{v}_e$  and obtain the final representation  $\mathbf{x}_e = \sigma(\mathbf{W}_2 \mathbf{v}_e + \mathbf{W}_3 \mathbf{h}_e)$ . Here,  $\parallel$  denotes concatenate operation.  $\sigma$  is non-linear function, and we use  $\tanh$ .  $\mathbf{u} \in \mathbb{R}^{d_e \times 1}$ ,  $b_1 \in \mathbb{R}$ ,  $\mathbf{W}_1 \in \mathbb{R}^{d_e \times 2d_e}$ ,  $\mathbf{W}_2 \in \mathbb{R}^{d_e \times d_e}$  and  $\mathbf{W}_3 \in \mathbb{R}^{d_e \times d_e}$  are learnable parameters. We pack the encoder output of each entity of  $s_q$  into matrix  $\mathbf{X}_q$ .  $d_e$  is the embedding dimension.

**Transformer Matching Processor** Transformer Matching Processor contains a stack of  $L$  identical blocks which mainly includes multi-head local-global attention module, position-wise feed-forward network (FFN) and layer normalization (LayerNorm). For  $\mathbf{X}_q$ , the multi-head local-global attention (MHA) can be denoted as  $\text{MHA}(\mathbf{X}_q) = \parallel_{i=1}^H (\text{Attn}_{\text{global}}(\mathbf{X}_q) + \text{Attn}_{\text{local}}(\mathbf{X}_q)) \mathbf{W}^O$ , where  $\text{Attn}_{\text{global}}(\cdot)$  and  $\text{Attn}_{\text{local}}(\cdot)$  indicate global attention and local attention function we describe later.  $H$  is the number of attention heads.  $\mathbf{W}^O \in \mathbb{R}^{Hd_e \times Hd_e}$  is learnable projection matrix. Fig. 1 illustrates the architectures of local and global attention. Please refer to (Vaswani et al. 2017) for more details about FFN and LayerNorm.

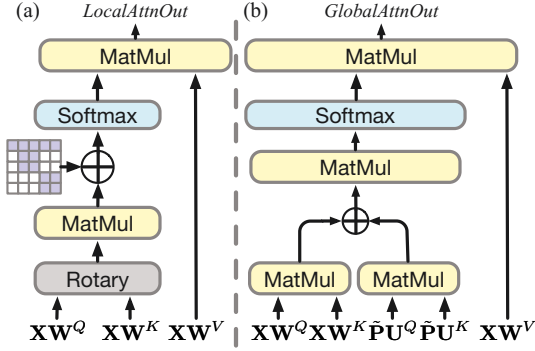


Figure 1: (a) Details of local attention; (b) Details of global attention. We sum *LocalAttnOut* and *GlobalAttnOut* and feed it to the rest structure of transformer.

**Local Attention** Firstly, we construct a block attention mask matrix  $\mathbf{M}_{\text{local}}$  to enable self-attention module capturing intra-triple entity interaction features. We further utilize rotary operation  $f_R^{\{Q,K\}}(e, m) = (\mathbf{W}^{\{Q,K\}} \mathbf{x}_e) e^{im\theta}$  for each entity  $e$  to endow entities with role information (head or tail) within triple, where  $m$  is 1 and 2 for head and tail, respectively. Finally, the local attention output is computed as  $\mathbf{H}_{\text{local}} = \text{Softmax}(\frac{f_R^Q(\mathbf{x}_q) \cdot f_R^K(\mathbf{x}_q)^T}{\sqrt{d}} + \mathbf{M}_{\text{local}}) \mathbf{x}_q \mathbf{W}^V$ .  $\mathbf{M}_{\text{local}}$  is obtained follows:

$$m_{i,j} = \begin{cases} i = 0 \text{ or } i = j \text{ or} \\ 0, & j = i + 1, i \in \{1, \dots, 2K + 1\} \text{ or} \\ & j = i - 1, i \in \{2, \dots, 2K + 2\}; \\ -\text{inf}, & \text{else} \end{cases} \quad (1)$$

**Global Attention** We first assign position  $i$  for both entity  $h_i$  and  $t_i$  of triple  $(h_i, t_i)$ , while the position for [CLS] and query pair  $(h_q, t_q)$  is set to 0 and  $K + 1$ , respectively. A  $d_e$  dimensional learnable embedding is applied to obtain representation of each position then we pack them into a matrix  $\tilde{\mathbf{P}}$ . Finally, we compute the global stage hidden state following  $\mathbf{H}_{\text{global}} = \text{Softmax}(\mathbf{A}_g) \mathbf{x}_q \mathbf{W}^V$ , where  $\mathbf{A}_g = \frac{(\mathbf{x}_q \mathbf{W}^Q)(\mathbf{x}_q \mathbf{W}^K)^T + (\tilde{\mathbf{P}} \mathbf{U}^Q)(\tilde{\mathbf{P}} \mathbf{U}^K)^T}{\sqrt{2d_e}}$ .

**Prediction and Optimization** Assuming the last hidden state of [CLS] is  $\mathbf{z}_{\text{CLS}}^L$ , we obtain final score  $\bar{s}_q$  of  $(h_q, t_q)$  follow  $\bar{s}_q = \text{Softmax}(\mathbf{U}_2^T (\mathbf{W}_4 \mathbf{z}_{\text{CLS}}^L))$ , in which  $\mathbf{W}_4 \in \mathbb{R}^{d_e \times Hd_e}$ ,  $\mathbf{U}_2^T \in \mathbb{R}^{d_e \times 2}$ ,  $\bar{s}_q \in \mathbb{R}^2$  is a 2-dimensional real vector while  $s_q^{(1)}$  indicates the probability of whether query  $(h_q, r, t_q)$  holds. We calculate loss  $\mathcal{L} = -\sum_{q \in \mathbb{S}^+ \cup \mathbb{S}^-} ((1 - y_q) \log(s_q^{(0)}) + y_q \log(s_q^{(1)}))$ , where  $\mathbb{S}^+$  and  $\mathbb{S}^-$  indicate positive and negative query sequence respectively.  $y_q \in \{0, 1\}$  is the sequence label (negative or positive).

## Experiments

**Datasets and Baselines** We use two public datasets NELL-One and Wiki-One proposed by GMatching(Xiong et al. 2018). We select Mean Reciprocal Rank (MRR), Hits@1 (H1) and Hits@10 (H10) as evaluation metrics. The

Model	NELL-One			Wiki-One		
	H1	H10	MRR	H1	H10	MRR
GMatching <sup>†</sup>	.119	.313	.185	.120	.336	.200
FSRL	.114	.294	.170	.091	.267	.149
FAAN	.116	.316	.187	.146	<b>.362</b>	.215
TransAM	<b>.152</b>	<b>.360</b>	<b>.225</b>	<b>.184</b>	.358	<b>.242</b>

Table 1: Experiment Results on NELL-One and Wiki-One. Best results are boldfaced. <sup>†</sup> indicates results are taken from the origin paper.

few-shot size  $K$  is set to 1, *i.e.*, one-shot. We compare TransAM with three baselines most relevant to our research, GMatching(Xiong et al. 2018), FSRL(Zhang et al. 2020) and FAAN(Sheng et al. 2020). Results of FSRL and FAAN are obtained using their official implementations with recommended hyperparameters reported in their papers.

**Implementation Details** We choose ComplEx pre-trained embedding vectors to initialize all entities and relations. Embedding dimensionality is 100 for NELL-One and 50 for Wiki-One. We set the number of Transformer layers to 3 and 4, and the number of heads to 4 and 8 for NELL-One and Wiki-One, respectively. We warm up the model at the first 10k steps by linearly increasing the learning rate to  $5e^{-5}$  for NELL-One and  $6e^{-5}$  for Wiki-One, then decreasing it linearly to 0 until the last step.

**Results and Conclusions** From table 1, we observe that TransAM outperforms all three baselines in MRR and Hits@1 on both two datasets. These results reveal that TransAM is appropriate for addressing few-shot issue in knowledge graph completion. Our future work may consider introducing sophisticated structural bias to the transformer for complex few-shot relations.

## Acknowledgments

This work is supported by Beijing Nova Program of Science and Technology (Grant No. Z191100001119031), Guangxi Key Laboratory of Cryptography and Information Security (No.GCIS202111). Yi Liang is supported by BUPT Excellent Ph.D. Students Foundation under grant CX2019136. Bo Cheng is the corresponding author.

## References

- Sheng, J.; Guo, S.; Chen, Z.; Yue, J.; Wang, L.; Liu, T.; and Xu, H. 2020. Adaptive Attentional Network for Few-Shot Knowledge Graph Completion. In *EMNLP*.
- Vaswani, A.; Shazeer, N. M.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is All you Need. .
- Xiong, W.; Yu, M.; Chang, S.; Guo, X.; and Wang, W. Y. 2018. One-Shot Relational Learning for Knowledge Graphs. In *EMNLP*.
- Zhang, C.; Yao, H.; Huang, C.; Jiang, M.; Li, Z.; and Chawla, N. 2020. Few-Shot Knowledge Graph Completion. In *AAAI*.