Joint Pre-Encoding Representation and Sturcture Embedding for Efficient and Low-Resource Knowledge Graph Completion

Chenyu Qiu[1,2], Pengjiang Qian[1,2*], Chuang Wang[1,2], Jian Yao[1,2], Li Liu[1,2,3], Wei Fang[1,2], Eddie-Yin-Kwee Ng[4]

[1]School of Artificial Intelligence and Computer Science, Jiangnan University
[2]Engineering Research Center of Intelligent Technology for Healthcare, Ministry of Education, Jiangnan University
[3]Big Data Center, Affiliated Hospital of Jiangnan University
[4]Nanyang Technological University
qianpjiang@jiangnan.edu.cn

Abstract

Knowledge graph completion (KGC) aims to infer missing or incomplete parts in knowledge graph. The existing models are generally divided into structure-based and description-based models, among description-based models often require longer training and inference times as well as increased memory usage. In this paper, we propose **P**re-**E**ncoded **M**asked **L**anguage **M**odel (PEMLM)[1] to efficiently solve KGC problem. By encoding textual descriptions into semantic representations before training, the necessary resources are significantly reduced. Furthermore, we introduce a straightforward but effective fusion framework to integrate structural embedding with pre-encoded semantic description, which enhances the model's prediction performance on 1-N relations. The experimental results demonstrate that our proposed strategy attains state-of-the-art performance on the WN18RR (MRR+5.4% and Hits@1+6.4%) and UMLS datasets. Compared to existing models, we have increased inference speed by 30x and reduced training memory by approximately 60%.

1 Introduction

Knowledge graph contains of a series of triples, depicting relations between entities in the form of graphs or triples. These entities can encompass real world entities such as objects, concepts, events, etc., while relations denote their connections and interactions. The composition format of a triple is (h,r,t), where h and t are head entity and tail entity, and r represents the relation connecting the two entities. Popular knowledge graphs include Freebase (Bollacker et al., 2008), WorldNet (Miller, 1995), and WikiData (Vrandečić and Krötzsch, 2014). Although these knowledge graphs are composed of massive amounts of real-world data, many parts are still missing. KGC aims to complete knowledge graph information by accurately predicting unknown relations through leveraging known entities and relations. A comprehensive knowledge graph can be further used for downstream tasks, such as question answer (Yasunaga et al., 2021; Han and Gardent, 2023), relation extraction (Zhang et al., 2019a), recommendation systems (Wang et al., 2019a; Hu et al., 2022), etc.

Existing KGC methods can be divided into two categories: Embedding-based model and description-based model. Embedding-based models aiming to reflect real triples as faithfully as possible in the corresponding real or complex vector space through learning embedding representations, modeling entities and relations by capturing the inherent structure of the graph. Description-based models use textual descriptions of entities and relations as information to predict missing entities.

Most description-based models using text directly as input, the training requires substantial memory and time due to the necessity of processing sufficiently long sequences, the inference speed may significantly increase as well. For example, KG-BERT requires a large amount of GPU memory and several days for inference over the entire knowledge graph, making training extremely challenging under limited resources. The primary cause of these issues is the substantial volume of text inputs, which aggravates the processing pressure on the model. In order to reduce the length of input, we use the pooling features from the last layer instead of the entire description text, as we assume that the pooling features can effectively represent the overall semantic information of description. Therefore, pre-encoding all descriptions with a pre-trained language model can effectively reduce resource consumption.

Besides, recently studies have shown that combining structural embedding and description representation can improve model performance. This

*Corresponding author: Pengjiang Qian.
[1]https://github.com/qiucy23/PEMLM-KGC

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 15257–15269
November 12-16, 2024 ©2024 Association for Computational Linguistics

type of method incorporates structural information into the description-based model. However, some joint-based models inappropriately employ descriptive features for structural processing, which may lead to the disruption of description semantic.

The main contributions of this paper are as follows:

1) We propose a pre-encoding processing method to construct the embedding layer of the main Triplet Encoder, which reduce the memory required for processing text by encoding long text description into pre-encoded representation.

2) Instead of employing the entity replacement strategy commonly employed by most models, we use Masked Language Model (MLM) as the training backbone, directly classified the encoding results when predicting missing entities. Simultaneously calculating the ranking of all candidate entities reduces time complexity from O(N) to O(1).

3) Unlike other models that directly structure semantic descriptions or feed structural embedding into language models, we utilize a learnable module to acquire the fusion representation. This framework named PEMLM-F enhances the forecasting capability of PEMLM for 1-N relations.

4) Extensive experiments have demonstrated that our approaches achieve state-of-the-art results on WN18RR and UMLS datasets. Compared to existing description-based models, our approach has significantly improved inference speed and reduced memory requirements.

The structure of this article is as follows. We first introduce the related work of the KGC algorithm in § 2. Details about our model and fusion framework will be reported in § 3. The experimental results and further analysis will be explained in § 4 and § 5, respectively. Finally, we summarize our research in § 6.

2 Related Work

Knowledge graph completion algorithm aims to learn knowledge from existing knowledge graph and predicts missing entities to complete it. TransE (Bordes et al., 2013) models triples as translation paradigm $h + r \approx t$. TransH (Wang et al., 2014) projects each relation onto a hyperplane through a matrix. TransR (Lin et al., 2015) establishes distinct entity and relation vector space to enhance modeling capabilities. RotatE (Sun et al., 2019b) projects entities and relations into complex space, defining relations as rotational transformations from head entities to tail entities. DistMult (Yang et al., 2014) restricts the relation matrix to be a diagonal matrix, greatly reducing the number of parameters but making it challenging to model asymmetric relations. CompleX (Trouillon et al., 2016) addresses complex relation modeling by incorporating complex space. QuatE (Zhang et al., 2019b) introduces hyper-complex space on top of CompleX. With the advancement of deep learning, convolutional neural network models (Dettmers et al., 2018; Zhang et al., 2022) and attention-based models (Wang et al., 2019b; Zhang et al., 2020a) have also been introduced for link prediction tasks.

The description-based model uses textual descriptions of entities and relations for link prediction. SSP (Xiao et al., 2017) simultaneously learns topic models and em-bedding models to enhance the correlation between descriptions and triples. KG-BERT (Yao et al., 2019) applies BERT to encode context in description for knowledge graph completion. MTL-KGC (Kim et al., 2020) and LP-BERT (Li et al., 2023) extends language model by introducing a multi-task framework for KGC. MLMLM (Clouatre et al., 2020) employ masked language model to encode missing entities. KEPLER (Wang et al., 2021b) constructs a unified pre-trained language model and knowledge graph embedding model, handling both natural language tasks and KGC tasks. KGT5 (Saxena et al., 2022) developed a link prediction method using the Sequence to Sequence framework. GHN (Qiao et al., 2023) innovatively uses hard negatives mining and contrastive learning framework for KGC.

In addition, there has been an increasing research emphasis on joint-based models that integrating structure embedding and description representation in recent years. Pretrain-KGE (Zhang et al., 2020b) replaces embedding vectors in embedding models with description representations processed by BERT. BLP (Daza et al., 2021) encode node descriptions in order to train relation vectors for embedding models. StAR (Wang et al., 2021a) introduces a Siamese-style hybrid framework and integrates the results of RotatE. LASS (Shen et al., 2022) fine-tunes BERT with the loss of embedding models. KGLM (Youn and Tagkopoulos, 2022) embeds structures into pre-trained language models and utilizes the language model to learn graph representations. JointSE (Wei et al., 2023) introduces an enhancement module on top of fusion to filter out irrelevant information from the text. KICGPT (Wei et al., 2024) uses structural knowledge as a knowledge prompt to guide LLM learning.

3 Method

In this section, we introduce PEMLM and its fusion version, PEMLM-F. We first introduce the problem formulation in §3.1. The model structure of PEMLM is proposed in §3.2, with the supplement of the fusion framework in §3.3.

3.1 Problem Formulation

A common representation of a knowledge graph consists of a set of triples, which can be defined as $\mathcal{G} = (h, r, t)$. Each entity $h, r \in \mathcal{E}$ is connected by a relation $r \in \mathcal{R}$, where $\mathcal{E}$ and $\mathcal{R}$ represent the sets of entities and relations, respectively. In the example (*population group, interacts, age group*), *'population group'* is the head entity, *'interacts with'* is the relation, and *'age group'* is the tail entity.

The aim of knowledge graph completion is to predict missing entities on triple. Under the widely ranking evaluation protocol, predicting the correct tail entity for $(h, r, ?)$ involves ranking all possible triples after replacing the missing part with each entity, and the same applies to predicting $(?, r, t)$. In this paper, we construct the inverse relations r^{-1} along with their corresponding description, and add the inverse triples (t, r^{-1}, h) to the knowledge graph $\mathcal{G}$ (Dettmers et al., 2018). Therefore, we merge predicting $(h, r, ?)$ and $(?, r, t)$ into only predicting $(h, r, ?)$.

3.2 Model Architeture of PEMLM

The model structure of PEMLM is shown in Figure 1. The Description Encoder and Triplet Encoder are initialized using the same language model $\text{BERT}_{\text{base}}$ (Devlin et al., 2018a), but the description encoder does not engage in gradient optimization.

For each entity $e \in \mathcal{E}$ or relation $r \in \mathcal{R}$, there is a corresponding description e_{des} or r_{des}. Each description is represented as a sentence, such as the description of entity named '*Gary_Rydstrom*' is '*Roger Rydstrom is an American sound designer and director.*', the description of relation named '*_has_part*' is '*has part*'. The sentence description is divided into token sequences by the tokenizer $(token_1, token_2, \ldots, token_n)$. The BERT model necessitates the addition of two special tokens, namely [CLS] and [SEP], which symbolize the beginning and the end of the sequence, correspondingly. Thus, the input sequence for each description of entity is $des_e = ([\text{CLS}], token^e_1, token^e_2, \ldots, token^e_n, [\text{SEP}])$. Similarly, the description of the relation is $des_r = ([\text{CLS}], token^r_1, token^r_2, \ldots, token^r_n, [\text{SEP}])$. By feeding des into the description encoder, the average pooling result of the final hidden layer is used to represent the semantic information of the relevant entity (or relation):

$$u = \text{MEAN}(\text{DesEnc}(des_e)) \tag{1}$$

As the representations of all entities and relations have been gathered, utilize these representations to create the embedding layer of the Triplet Encoder:

$$\begin{aligned} E &= [u^e_1, u^e_2, \ldots, u^e_n] \\ R &= [u^r_1, u^r_2, \ldots, u^r_m] \\ \text{Embedding} &= [E; R] \end{aligned} \tag{2}$$

where u^e_i denotes the semantic representation vector of the i-th entity, u^r_j indicates the semantic representation vector of the j-th relation. n and m refer to the quantities of entities and relation, respectively. Embedding is the constructed embedding layer.

Given a triple (h, r, t), we combine the head with relation representations and mask the entity that need to be predicted. The BERT model necessitate to understand the relative positions of each element in the input sequence to effectively capture the structure and relations within the sequence. By establishing position embedding to the inputs at each position, we can provide the model with information about the relative positions of entities and relations. This allows the model to gain a better understanding of the relations between different elements in the input sequence. Thus, we create the input sequence like

$$u^{input} = [e^0_{[\text{CLS}]}, u^1_h, u^2_r, e^3_{[\text{MASK}]}, e^4_{[\text{SEP}]}] \tag{3}$$

where $u^i_{\text{token}} = u_{\text{token}} + x^i_{\text{pos}}$, u_{token} represents the embedding layer representation of the corresponding token, x^i_{pos} represents the position embedding of the corresponding position. u_h and u_r denote pre-encoded descriptive semantic representations, respectively. For $e^i_{\text{SPE\_TOKEN}} = e_{\text{SPE\_TOKEN}} + x^i_{\text{pos}}$, where $e_{\text{SPE\_TOKEN}}$ indicates special token embedding representation of [CLS], [MASK], and [SEP]. Figure 2 illustrates the intuitive form.

To predict the entity being masked, we feed the u^{input} into the Triplet Encoder. We then treat the representation of the mask's location on the last

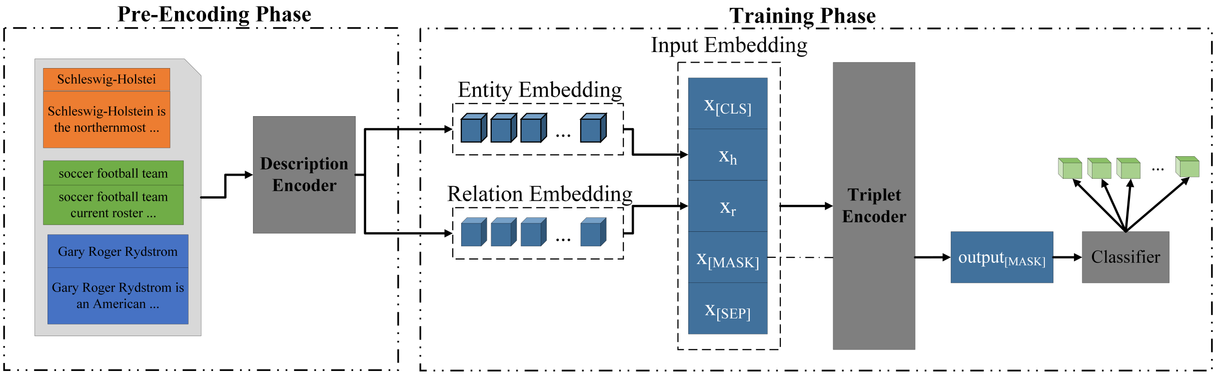

Figure 1: An overview of the PEMLM for link prediction. PEMLM consists of a pre-trained Description Encoder and a Triplet Encoder. The pre-Encoding and training stages can be conducted independently.

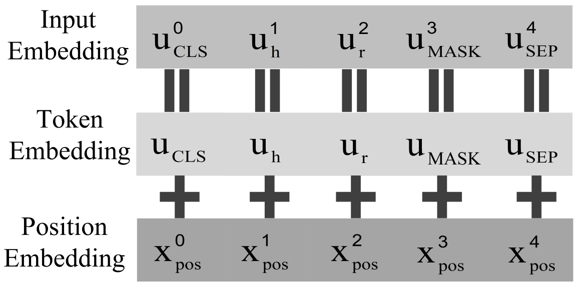

Figure 2: Composition of input embedding.

hidden layer as the encoded output for the entity that needs to be predicted:

$$\text{output}_{\text{MASK}} = \text{TripletEnc}(u^{input})[p_m, :] \tag{4}$$

where p_m is the positional index of the MASK token.

The task of prediction $(h, r, ?)$ can be viewed as a multi-classification issue for encoding the results of mask. We employ a dense layer as the classification layer, with the output dimension corresponding to the total number of entities. We then derive the classification probability using the softmax function

$$\tilde{t} = softmax(\text{output}_{\text{MASK}} \cdot W_c^{\top} + b^c) \tag{5}$$

where W_c and b_c are the learnable weights and biases of dense layer, respectively.

During the training phase, we utilize mul-ti-class cross-entropy loss to optimize the learnable parameters:

$$\mathcal{L}_{ce} = \sum_{i=1}^{n} y_i \log(\tilde{t_i}) \tag{6}$$

where y_i represents the true label of entity i, and $\tilde{t_i}$ denotes the likelihood of entity i being selected.

3.3 Structure Embedding Fusion

Embedding-based models have been extensively employed in the area of knowledge graph completion. Our research incorporates the concepts of the prominent translation model TransE (Bordes et al., 2013), which is also employed to enhance the embedding information of some joint-based models (Zhang et al., 2020b; Shen et al., 2022).

The principle of the translation model is manifested as $h + r \approx t$, with the goal of aligning the head entity with the tail entity in the same semantic space through relations. The values of each element in the entity embeddings and relation embeddings will be set within the range of $[-a, a]$ during initialization:

$$a = gain \times \sqrt{\frac{6}{dim}} \tag{7}$$

$gain$ is the factor used for scaling, whereas dim represents the dimension of the embedding.

The evaluation function for the translation model is cosine similarity:

$$v_{hr} = |v_h + v_r| \tag{8}$$

$$f(h, r, t) = cosine(v_{hr}, v_t) = \frac{v_{hr} \cdot v_t}{\|v_{hr}\| \cdot \|v_t\|} \tag{9}$$

among them, v_h, v_r, and v_t denote the embedding representations of the head entity, relation, and tail entity in the translation model. $||$ and $\|\|$ symbolizes one norm and two normal forms, respectively. A higher score of $f(h, r, t_i)$ indicates a higher likelihood that the embedding model will forecast the entity i as the target tail on $(h, r, ?)$.

The fusion module takes as input a pre-encoded description representation, denoted as u, and a structural embedding, denoted as v. In our research,

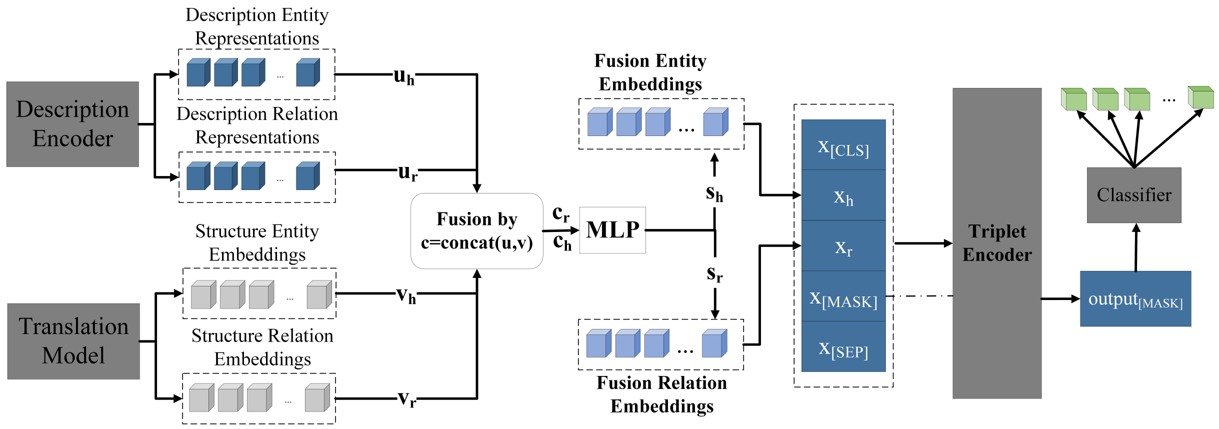

Figure 3: The architecture of PEMLM-F. u and v deonote description representation and structure embedding, respectively. c is the fusion representation generated from fusion module.

we integrate u and v, and then feed the combined expression into a trainable MLP. The fusion framework is shown as Figure 3. Through the procedure of optimizing the training of the model, the model will acquire the ability to learn appropriate fusion representation:

$$c(u, v) = Concat(u, v) \tag{10}$$

$$s = c(u, v) \cdot W_f^\top + b_f \tag{11}$$

the weights and biases of the fused MLP are denoted as W_f and b_f. It should be mentioned that u, v, and s all have the same dimensions.

To obtain the fusion expression, we shall first generate the input sequence

$$s^{input} = [e^0_{\text{[CLS]}}, s^1_h, s^2_r, e^3_{\text{[MASK]}}, e^4_{\text{[SEP]}}] \tag{12}$$

where $e^i_{\text{SPE\_TOKEN}} = e_{\text{SPE\_TOKEN}} + x^i_{\text{pos}}$, and $s^i_{\text{token}} = s_{\text{token}} + x^i_{\text{pos}}$. s_{token} denotes the fusion representation of the token.

We employ contrastive loss to optimize translation model so that it can learn about graph structures:

$$\mathcal{L}_{tr} = -\log \frac{e^{f(h,r,t)}}{e^{f(h,r,t)} + \sum_{i=1}^{|\mathcal{N}|} e^{f(h,r,t_i)}} \tag{13}$$

where $\mathcal{N}$ is the negative sampling entity set. The translation model may be forced by contrastive loss to learn embedding representations in which positive samples outperform negative samples.

The final total loss of the entire model requires weighting the classification losses $\mathcal{L}_{ce}$ and contrastive loss $\mathcal{L}_{tr}$:

$$\mathcal{L}_{total} = \mathcal{L}_{ce} + \alpha \cdot \mathcal{L}_{tr} \tag{14}$$

α is the loss weight parameter. All parameters, with the exception of the Description Encoder, are optimized by $\mathcal{L}_{total}$.

4 Experience

In §4.1, we will provide an overview of the experiment detailed setup. In §4.2 we will focus on the specific results and analysis.

4.1 Experiment Setup

Datasets. Our experiment will be conducted on three well-used benckmark datasets: FB15k-237 (Toutanova and Chen, 2015), WN18RR (Dettmers et al., 2018), and UMLS (Dettmers et al., 2018). For further details on datasets, please refer to Appendix F.

Baselines. We will compare our methods on the embed-ding-based model, description-based model, and joint-based model. Embedding-based models include translation models TransE (Bordes et al., 2013), RotatE (Sun et al., 2019b), and REP-OTE (Wang et al., 2022a); tensor decomposition models such as DistMult (Yang et al., 2014) and Tucker (Balažević et al., 2019); convolutional neural network model ConvE (Dettmers et al., 2018) and representation learning model CKRL (Sabet et al., 2023). Description-based models contain KG-BERT (Yao et al., 2019), multi-task learning models including MTL-KGC (Kim et al., 2020) and LP-BERT (Li et al., 2023), and mask language based model MLMLM (Clouatre et al., 2020). Joint-based model involve models that structure semantic representation like Pretrain-KGE (Zhang et al., 2020b) and LASS (Shen et al., 2022), ensemble embedding model like StAR (Wang et al., 2021a), contextualize structural embedding as KGLM (Youn

Models	FB15k-237				WN18RR				UMLS	
	MRR↑	Hits1↑	Hits3↑	Hits10↑	MRR↑	Hits1↑	Hits3↑	Hits10↑	MR↓	Hits10↑
Embedding-based models										
TransE (Bordes et al., 2013)⋆	0.279	0.198	0.376	0.441	0.243	0.043	0.441	0.543	1.84	0.989
DistMult (Yang et al., 2014)⋆	0.281	0.199	0.301	0.446	0.444	0.412	0.470	0.504	5.52	0.846
ConvE (Dettmers et al., 2018)⋆	0.312	0.225	0.341	0.497	0.456	0.419	0.470	0.531	1.51	0.990
RotatE (Sun et al., 2019b)⋆	0.338	0.241	0.375	0.533	0.476	0.428	0.492	0.571	-	-
Tucker (Balažević et al., 2019)⋆	0.358	0.266	0.394	0.544	0.470	0.443	0.482	0.526	-	-
REP-OTE (Wang et al., 2022a)	0.354	0.262	0.388	0.540	0.488	0.439	0.505	0.588	-	-
CKRL (Sabet et al., 2023)	**0.416**	**0.279**	-	**0.594**	0.501	0.445	-	0.595	-	-
Description-based models										
KG-BERT (Yao et al., 2019)†	0.237	0.144	0.260	0.427	0.219	0.095	0.243	0.497	1.47	0.990
MTL-KGC (Kim et al., 2020)†	0.267	0.172	0.298	0.458	0.331	0.203	0.383	0.597	-	-
MLMLM (Clouatre et al., 2020)	-	-	-	-	0.502	0.439	0.542	0.610	-	-
LP-BERT (Li et al., 2023)‡	0.310	0.223	0.336	0.490	0.482	0.343	0.563	**0.752**	1.17	0.995
Joint-based models										
Pretrain-KGE (Zhang et al., 2020b)	0.350	0.250	0.384	0.554	0.488	0.437	0.509	0.586	-	-
$\text{StAR}_{\text{BERT-base}}$ (Wang et al., 2021a)⋆	0.362	0.264	0.399	0.559	0.364	0.222	0.436	0.647	1.49	0.991
$\text{LASS}_{\text{BERT-base}}$ (Shen et al., 2022)	-	-	-	-	0.479	-	-	0.725	1.39	0.991
KGLM (Youn and Tagkopoulos, 2022)	0.289	0.200	0.314	0.468	0.467	0.330	0.538	0.741	1.19	0.995
PEMLM (w/o fusion)	0.339	0.249	0.370	0.520	0.545	0.490	0.565	0.609	1.40	0.992
PEMLM-F (w/ fusion)	0.355	0.264	0.389	0.538	**0.556**	**0.509**	**0.573**	0.648	**1.14**	**0.997**

Table 1: Link prediction results on FB15k-237, WN18RR and UMLS. ⋆ results are taken from (Wang et al., 2021a). † results are taken from (Kim et al., 2020). ‡ results are generated by our paper. Others are taken from the original papers. PEMLM-F denotes PEMLM with structure embedding fusion.

and Tagkopoulos, 2022).

Evaluation metrics. In the evaluation stage of link prediction, we ignore other potentially correct tails of the test samples $(h, r, ?)$ in the training set, validation set, and test set, based on the filtered strategy (Bordes et al., 2013). We then determine the ranking of the correct tail entity among all candidate entities. In our study, we use the softmax value from classifier as score to rank entities. The obtained rank will be used to calculate the following evaluation metrics: Mean Rank (MR), Mean Reciprocal Rank (MRR), and Hits@K. MR is computed by taking the average of all ranks, whereas is computed by taking the average of the reciprocals of all rankings. Hits@K compute the sum of $rank <$ K, where K $\in \{1, 3, 10\}$

We choose MRR as the main evaluation metric (MR on UMLS), save the model with the best results obtained on the validation set, and demonstrate the final result on the test set.

Hypermeter settings. We employ BERT-base for the initialization of both the Description Encoder and Triple Encoder. The Description Encoder obtains a maximum text length of 128. The feature dimension of the embedding model is set to be the same as the embedding layer dimension of the BERT-base model, with both being 768. The embedded model on the FB15k-237 and WN18RR datasets uses a negative sampling size of 2048 ($|\mathcal{N}| = 2048$), while on the UMLS dataset, the negative sampling size is 134 ($|\mathcal{N}| = 134$). We set learning rates on the Fb15k-237, WN18RR, and UMLS datasets, respectively $lr = 10^{-5}, 3 \times 10^{-5}$, and 10^{-5}. Based on extensive experiments, we believe that WN18RR is more susceptible to the influence of embedded models compared to FB15k-237 (mentioned on Appendix B), therefore α in Eq.14 set to 2.0 on WN18RR and 1.0 on FB15k-237/UMLS, respectively. Moreover, we build the inverse relations of FB15k-237 through using '$/be$' and apply '$be\_$' to WN18RR and UMLS. The model was trained using a batch size of 256 for 50 (150 on UMLS) epochs. Our experiment runs on RTX 3080Ti 12G.

4.2 Experiment Results and Analysis

We present the main link prediction results of PEMLM in Table 1. In general, our approach has been shown to achieve state-of-the-art performance on UMLS datasets and perform well on most evaluation metrics on WN18RR datasets.

Among all models, our model outperforms existing metrics in Hits@1 and Hits@3, with improvements of 6.4% and 3.1% respectively. Compared to the Joint-based model, our model has improved by 7.2% and 6.4% in Hits@1 and Hits@3 respectively.

Fusion Way	MRR	Hits1	Hits3	Hits10
$c_1 = concat(u, v)$	0.556	0.509	0.573	0.648
$c_2 = u + v$	0.554	0.505	0.571	0.644
$c_3 = concat(u, u \times v, u - v, v)$	0.550	0.503	0.565	0.642

Table 2: Comparison results of different fusion ways on WN18RR.

Although Hits@10 are lower than the current best model, the average performance of PEMLM is still the best, with an improvement of 5.4% and 6.8% in the MRR of all models and joint-based models, respectively. We believe that adopting multi-classification loss will make the model focused on the target entity, resulting in improved high ranking results (Hits@1 and Hits@3). However, when the model encounters missing triple that may indicate many tail entities, prediction becomes more difficult for it. The optimal outcome is to have all potential correct tail entities placed in the top-ranked portion, so that for each triple to achieve a higher ranking during evaluation. Nevertheless, there are circumstances where model optimization may occur challenges, bring about ignoring specific triples in Triple$(h, r, ?) = \{(h, r, t_1), (h, r, t_2), \ldots, (h, r, t_p)\}$, thus affecting the Hits@10 metric.

The overall results on the FB15k-237 dataset indicate that the performance of both description-based model and joint-based model are inferior than that of embedding-based model on FB15k-237. We infer that the first possible reason is the significantly higher average node degree of FB15k-237 compared to WN18RR (59.7 vs 4.3), suggesting a potentially more sophisticated graph structure. Secondly, the FB15k-237 dataset has significantly longer description texts compared to WN18RR. On average, the length of entity descriptions in FB15k-237 reaches 864. Most description-based and joint-based models have a maximum text length of 128, which might result in inadequate learning of description semantics owing to text truncation.

5 Discussion

In this section, we will investigate four questions: How to Pre-Encode a good description representation (§5.1), how to properly fuse structure embeddings with description representations (§5.2), why we describe PEMLM as Efficient and Low-Resource (§5.3), and the performance of PEMLM on different type of relations (§5.4).

Pooling	MRR	Hits1	Hits3	Hits10
CLS-Pool	0.531	0.490	0.565	0.609
Mean-Pool	0.545	0.502	0.557	0.627
Max-Pool	0.480	0.449	0.489	0.540

Table 3: Performance of different pooling methods.

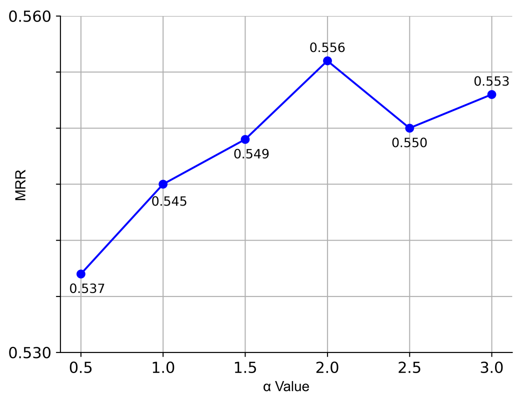

Figure 4: MRR on WN18RR with different α.

5.1 Semantic Representation Analysis

The description representation produced by the Description Encoder will have an effect on subsequent prediction results. CLS pool, mean pool, max pool, and attention are frequently employed as semantic representations in sentences. The CLS token utilizes the first token of the last hidden state as the representation. Mean pool applies average pooling to the last hidden state of each word, while max pool applies maximum pooling.

Table 3 presents a comparison of results from three pooling strategies used to WN18RR. Mean Pooling provides superior semantic representation quality, exhibiting a 6.5% enhancement in MRR as compared to max pool.

5.2 Fusion Way Analysis

Loss Ratio. The overall optimization of PEMLM-F is influenced by the loss weight α while training the complete model using $\mathcal{L}_{total}$. We select the value of α from the interval $[0.5, 3.0]$ with increments of 0.5. When α is greater than 1, the optimizer will place more attention on the embedding model. Figure 4 demonstrates that the model achieves best performance when $\alpha = 2$.

Fusion Way. To determine the fusion approach that offers better fusion expression, we conducted a comparison of three fusion strategies:

(1) $c_1 = concat(u, v)$. This method concatenates two vectors along their dimensions, forming a larger vector. The concatenated fusion features allow the fusion module to learn and discover the relationships between u and v. This method can retain and learn from all the information in the original features, but the high dimensionality of the concatenated features may increase computational complexity and the risk of overfitting.

(2) $c_2 = u + v$. This method adds two vectors by feature is a linear combination based on feature level. We set the lengths of u and v to be the same, so the length of the fused vector is the same as u and v. The advantage of this method is its low computational cost, as it does not require additional feature space. However, it only captures the features after adding u and v together, without considering the interactions or independent information between u and v.

(3) $c_3 = concat(u, u \times v, u - v, v)$. Proposed by StAR (Wang et al., 2021a), this method concatenates multiple representations of features, capturing richer feature information. However, the feature dimensionality increases significantly, making it more challenging for the model to learn. From Table 2, we can infer that the performance of c3 is slightly lower than c1 and c2, which we speculate is due to its high dimensionality. The better performance of c2 compared to c1 also demonstrates that concatenating complete features provides more fusion information than simple addition.

5.3 Resource Consumption Analysis

To highlight the low-resource and efficiency of PEMLM, we conduct a comparative analysis with KG-BERT, StAR, and SimKGC (Wang et al., 2022b) in terms of the time and memory resources needed for training and inference. For fairness, all training batch sizes are uniformly set to 32, while the inference batch size is set to 1 for evaluation. Table 4 shows our approaches exhibit significantly improved performance in terms of inference time. It should be noted that the time required for pre-encoding only needs 7 minutes, and pre-encode embedding of the same dataset can be saved and reuse. Our total training time was 4.2 hours and 5.8 hours, respectively, which was much less than the training times of KG-BERT (10 hours) and StAR (10.5 hours). Due to the use of a contrastive learning framework in SimKGC, its performance is influenced by the input batch size. When the input batch size is restricted (simulating a resource-constrained environment), our method still demonstrates a relative advantage. The reduction in time consumption is mainly due to the fact that when predicting entities, the probabilities of all possible entities are obtained at once, instead of replacing each possible entity sequentially, reducing the time complexity of predicting entities from O(N) to O(1). In addition, the memory required for training our model is much less than that of other models, allowing for the training of large-scale graph datasets under resource-limited conditions. We believe that the pre-encoding method is key to reducing memory usage. For example, when processing the textual descriptions of WN18RR, the Description Encoder com-presses the entity description texts, which have an average token length of 89.8, into semantic representation with a token length just one, thereby reducing the memory required for text representation.

Model	Time		Memory		MRR
	T/EP	Infer	Train	Infer	
KG-BERT	120m	4day	8.5G	12G	0.219
StAR	90m	30m	9.6G	2.4G	0.364
SimKGC	11m	3m	6.2G	4.0G	0.543
PEMLM	**5m**	**1m**	**3.6G**	**2.0G**	0.545
PEMLM-F	7m	1.2m	4.4G	2.3G	**0.556**

Table 4: Comparisons of the time, memory, and MRR required for training and inference with KG-BERT and StAR on WN18RR. T/EP denotes a training epoch.

Model	1-1	1-N	N-1	N-N
PEMLM	0.976	0.091	0.468	0.948
PEMLM-F	0.976	0.127	0.460	0.949

Table 5: MRR performance of PEMLM and PEMLM-F on different types of relations on WN18RR.

5.4 Fine-Grain Analysis

In complex knowledge graphs, the types of relations are categorized as 1-1, 1-N, N-1, and N-N, as defined in (Bordes et al., 2013). For example, 'verb group' is the 1-1 relation type, 'has part' is the 1-N relation type, 'Hypernym' is the N-1 relation type, and 'also see' is the N-N relation type. It should be noted that we have constructed inverse relations, accordingly 'be has part' is the N-1 relation type. The results in Table 5 show that after incorporating structural embedding information, the model's performance on 1-N type relations significantly improved.

6 Conclusion

In this paper, we introduce a pre-encoded masking language model for efficient knowledge graph completion. By pre-encoding semantic representations extracted from description texts, we significantly reduce the memory requirement and inference time. To integrate structural and semantic information effectively, we build a learnable fusion module that integrates the information of both representations. Experiment results demonstrate that our PEMLM-F surpasses the majority of baseline metrics on the WN18RR dataset. Overall, the model achieves state-of-the-art performance on both UMLS and WN18RR datasets, while significantly enhancing inference speed and reducing memory consumption.

Limitations

Although our work performs better than existing models, it still has the following limitations:
1) Dataset limitations. Similar to all description-based and joint-based models, the dataset we used must contain descriptions of entities and relations. Additionally, the quality of the description text also affects the performance of our model. Therefore, when the dataset lacks specific descriptions or uses only names as descriptions, the model may not perform as well as expected.
2) Methodological Limitations. Recent research on embedding-based models has mainly focused on extending the modeling capabilities to complex domain spaces. It's challenging to align embeddings from the complex domain with pre-encoded representations in PEMLM-F. We will explore how to address this issue in future work.

Ethics Statement

This study is conducted with full compliance with the ethical code set out in the ACL Code of Ethics.

The datasets used in this study are all publicly available datasets for research. The pre-trained BERT model weights are sourced from *huggingface*. We use it in accordance with the Apache-2.0 license.

Knowledge graph completion technology may be used for reasoning and completion applications in knowledge graphs such as common sense and healthcare. However, the efficacy of automated completion may be compromised by issues such as outdated or biased data, potentially leading to erroneous or misleading inferences. When using knowledge graph completion techniques to infer unknown nodes, it is necessary to be warned of potential risks and possible consequences, and may require additional review of the predicted results. Furthermore, the application of completion techniques should be carefully managed to avoid speculative use concerning personal privacy or discriminatory decision-making.

Acknowledge

This work was supported in part by the National Natural Science Foundation of China under Grant 62172192, U20A20228, and 62171203, and in part by the Science and Technology Demonstration Project of Social Development of Jiangsu Province under Grant BE2019631.

References

Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Louis Clouatre, Philippe Trempe, Amal Zouaq, and Sarath Chandar. 2020. Mlmlm: Link prediction with mean likelihood masked language model. *arXiv preprint arXiv:2009.07058*.

Daniel Daza, Michael Cochez, and Paul Groth. 2021. Inductive entity representations from text via link prediction. In *Proceedings of the Web Conference 2021*, pages 798–808.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Kelvin Han and Claire Gardent. 2023. Generating and answering simple and complex questions from text and from knowledge graphs. In *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 285–304.

Bojing Hu, Yaqin Ye, Yingqiang Zhong, Jiao Pan, and Maosheng Hu. 2022. Transmkr: Translation-based knowledge graph enhanced multi-task point-of-interest recommendation. *Neurocomputing*, 474:107–114.

Weiqiang Jin, Biao Zhao, Yu Zhang, Jia Huang, and Hang Yu. 2024. Wordtransabsa: Enhancing aspect-based sentiment analysis with masked language modeling for affective token prediction. *Expert Systems with Applications*, 238:122289.

Bosung Kim, Taesuk Hong, Youngjoong Ko, and Jungyun Seo. 2020. Multi-task learning for knowledge graph completion with pre-trained language models. In *Proceedings of the 28th international conference on computational linguistics*, pages 1737–1743.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Da Li, Boqing Zhu, Sen Yang, Kele Xu, Ming Yi, Yukai He, and Huaimin Wang. 2023. Multi-task pre-training language model for semantic network completion. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 22(11):1–20.

Pengfei Li, Liangyou Li, Meng Zhang, Minghao Wu, and Qun Liu. 2022. Universal conditional masked language pre-training for neural machine translation. *arXiv preprint arXiv:2203.09210*.

Xiaobo Liang, Zecheng Tang, Juntao Li, and Min Zhang. 2023. Open-ended long text generation via masked language modeling. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 223–241.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 29.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Zile Qiao, Wei Ye, Dingyao Yu, Tong Mo, Weiping Li, and Shikun Zhang. 2023. Improving knowledge graph completion with generative hard negative mining. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5866–5878.

Maryam Sabet, MohammadReza Pajoohan, and Mohammad R Moosavi. 2023. Representation learning of knowledge graphs with correlation-based methods. *Information Sciences*, 641:119043.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. *arXiv preprint arXiv:2203.10321*.

Jianhao Shen, Chenguang Wang, Linyuan Gong, and Dawn Song. 2022. Joint language semantic and structure embedding for knowledge graph completion. *arXiv preprint arXiv:2209.08721*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019a. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

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

Weidong Tian, Haodong Li, and Zhong-Qiu Zhao. 2022. Dual capsule attention mask network with mutual learning for visual question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5678–5688.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, pages 57–66.

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

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*, 57(10):78–85.

Bo Wang, Tao Shen, Guodong Long, Tianyi Zhou, Ying Wang, and Yi Chang. 2021a. Structure-augmented text representation learning for efficient knowledge graph completion. In *Proceedings of the Web Conference 2021*, pages 1737–1748.

Huijuan Wang, Siming Dai, Weiyue Su, Hui Zhong, Zeyang Fang, Zhengjie Huang, Shikun Feng, Zeyu Chen, Yu Sun, and Dianhai Yu. 2022a. Simple and effective relation-based embedding propagation for knowledge representation learning. *arXiv preprint arXiv:2205.06456*.

Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. 2022b. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4281–4294.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019a. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958.

Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019b. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 950–958.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2021b. Kepler: A unified model for knowledge embedding and pre-trained language representation. *Transactions of the Association for Computational Linguistics*, 9:176–194.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI conference on artificial intelligence*, volume 28.

Xiao Wei, Yunong Zhang, and Hao Wang. 2023. Joint semantic embedding with structural knowledge and entity description for knowledge representation learning. *Neural Computing and Applications*, 35(5):3883–3902.

Yanbin Wei, Qiushi Huang, James T Kwok, and Yu Zhang. 2024. Kicgpt: Large language model with knowledge in context for knowledge graph completion. *arXiv preprint arXiv:2402.02389*.

Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. 2017. Ssp: semantic space projection for knowledge graph embedding with text descriptions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

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

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *arXiv preprint arXiv:1909.03193*.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*.

Jason Youn and Ilias Tagkopoulos. 2022. Kglm: Integrating knowledge graph structure in language models for link prediction. *arXiv preprint arXiv:2211.02744*.

Ningyu Zhang, Shumin Deng, Zhanlin Sun, Guanying Wang, Xi Chen, Wei Zhang, and Huajun Chen. 2019a. Long-tail relation extraction via knowledge graph embeddings and graph convolution networks. *arXiv preprint arXiv:1903.01306*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019b. Quaternion knowledge graph embeddings. *Advances in neural information processing systems*, 32.

Zhanqiu Zhang, Jie Wang, Jieping Ye, and Feng Wu. 2022. Rethinking graph convolutional networks in knowledge graph completion. In *Proceedings of the ACM Web Conference 2022*, pages 798–807.

Zhao Zhang, Fuzhen Zhuang, Hengshu Zhu, Zhiping Shi, Hui Xiong, and Qing He. 2020a. Relational graph neural network with hierarchical attention for knowledge graph completion. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 9612–9619.

Zhiyuan Zhang, Xiaoqian Liu, Yi Zhang, Qi Su, Xu Sun, and Bin He. 2020b. Pretrain-kge: learning knowledge representation from pretrained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 259–266.

Ran Zhou, Xin Li, Ruidan He, Lidong Bing, Erik Cambria, Luo Si, and Chunyan Miao. 2021. Melm: Data augmentation with masked entity language modeling for low-resource ner. *arXiv preprint arXiv:2108.13655*.

A Evaluation Metrics Iteration

The evaluation metrics for PEMLM-F with epoch validation on three datasets are shown in the Figure 5. The MR metric on the WN18RR and FB15k-237 datasets has increased after 15/20 epoch, while other indicators still maintain a convergence trend. This is because the classification model focuses more on predicting high rankings.

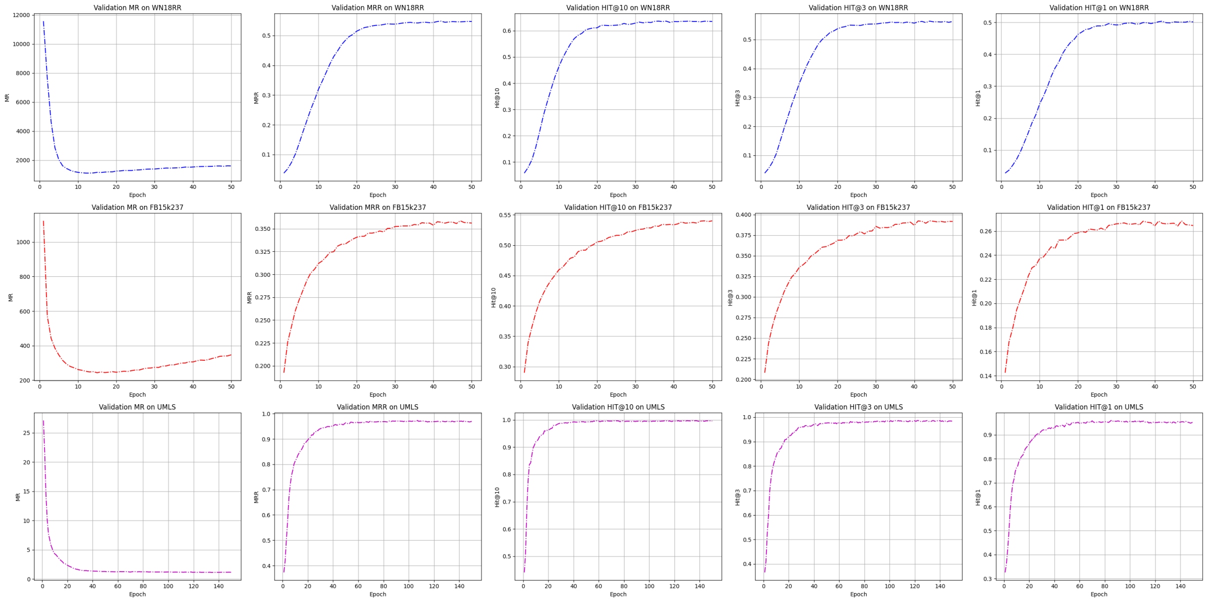

Figure 5: The training loss and validation metrics of PEMLM-F on three datasets.

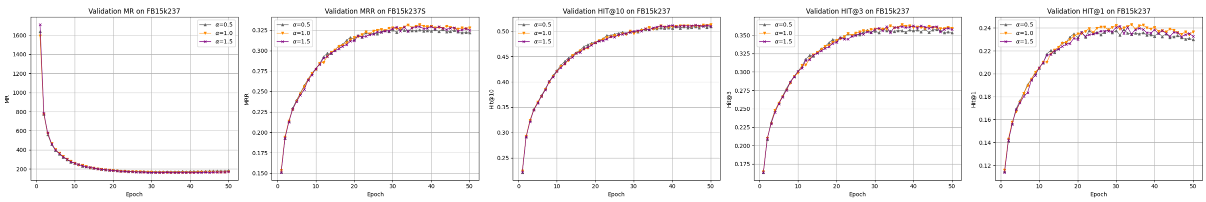

Figure 6: The metrics on FB15k237 with different α.

B Fusion parameter on FB15k237

Figure 6 shows the validation metrics of PEMLM-F on the FB15K237 dataset with different α on eq.14. The model obtained the best metric when $\alpha = 1.0$, and a larger α actually leads to poorer results.

C Influence of position encoding

	MRR	Hits1	Hits3	Hits10
w/o position encoding	0.529	0.484	0.547	0.618
w/ position encoding	**0.545**	**0.502**	**0.557**	**0.627**

Table 6: Performance comparison with and without position encoding on WN18RR

In order to demonstrate the supplementary information of position encoding on the relative position of triples, we conducted ablation experiments on the WN18RR dataset using PEMLM. The results in Table 6 show that the relative position information of triples provided by position encoding is able to improve the link prediction ability of model.

D Case Study

In order to visually display the results of PEMLM-F and PEMLM in link prediction, we selected one relation from each of the four types of relations. The sign under the relation on Table 7 represents the type of the relation. We predict missing entities on the test set of WN18RR and select the top-5 ranking entities. On these four sets of examples, The ranking results of PEMLM-F is superior to that of PEMLM, which proves the improvement in inference performance brought by the fusion architecture.

E Mask Language Model

Mask Language Model (MLM) is a subtask in natural language processing (Devlin et al., 2018b), which is used in the pre-training process of many models (Liu et al., 2019; Lan et al., 2019; Sun et al., 2019a). This method uses meaningless mask token to represent randomly masked words in text sequences and requires the model to predict the masked positions. Language model is capable of understanding contextual sequence information

Table 7: Example of PEMLM and PEMLM-F prediction results of top-5 ranking on the test set of WN18RR dataset. The groundtruth is in bold font. We select one relation from all categories as examples.

Relation	Triple Example	PEMLM-F	PEMLM
has part (1-N)	(africa, has part, **senegal**)	2,(eritrea, **senegal**, madagascar, tunisia, morocco)	2,(eritrea, **senegal**, tunisia, algeria, vincent)
similiar to (1-1)	(clean, similiar to, **rigidify**)	1,(**rigidify**, cleanness, tidy, antiseptic, antiseptic)	1,(**rigidify**, coloured, antiseptic, attractive, clean)
hypernym (N-1)	(disapproval, hypernym, **substance**)	3,(activity, mental object, **substance**, state, status)	9,(ire, vexation, dread, speech act, modification)
also see (N-N)	(travel,also see,**progress**)	1,(**progress**, proceed, give way, come up, pass)	6,(zip, surface, come up, slither, give way)

and learning intrinsic connections by predicting the masked words. Benefiting from the flexibility of masking, the MLM framework can be fine-tuned based on PLMs for various tasks, such as sentiment analysis (Jin et al., 2024), machine translation (Li et al., 2022), QA system (Tian et al., 2022), name entity recognition (Zhou et al., 2021), and text generation (Liang et al., 2023).

Dataset	FB15k-237	WN18RR	UMLS
Entities	14541	40943	135
Relations	237	18	46
Train	272115	86835	5216
Dev	17535	3034	652
Test	20466	3134	661

Table 8: Detailed scale of three benchmark datasets

F Details of datasets

The scale of benchmark datasets is shown as Table 8. With filtering to avoid information leaking, the FB15k-237 and WN18RR are built from the FB15k and WN18 databases (Bordes et al., 2013). FB15k is a portion of the FreeBase knowledge base, whereas WN18 is a portion of the WordNet knowledge bank. Freebase is a well organised database that encompasses the interconnectedness of many entities around the world. WordNet is a dataset that provides a detailed description of the relations between English words, including both symmetric and asymmetric interactions, as well as combinations of these relations. UMLS dataset is a compact knowledge graph designed for the domains of biomedical and health.