Multi-task Pre-training Language Model for Semantic Network Completion

DA LI, Tencent, China
BOQING ZHU, National Key Lab of Parallel and Distributed Processing, National University of Defense Technology, China
SEN YANG, Bioinformatics Center of AMMS, China
KELE XU, National Key Lab of Parallel and Distributed Processing, National University of Defense Technology, China
MING YI and YUKAI HE, Tencent, China
HUAIMIN WANG, National Key Lab of Parallel and Distributed Processing, National University of Defense Technology, China

Semantic networks, exemplified by the knowledge graph, serve as a means to represent knowledge by leveraging the structure of a graph. While the knowledge graph exhibits promising potential in the field of natural language processing, it suffers from incompleteness. This article focuses on the task of completing knowledge graphs by predicting linkages between entities, which is fundamental yet critical. Traditional methods based on translational distance struggle when dealing with unseen entities. In contrast, semantic matching presents itself as a potential solution due to its ability to handle such cases. However, semantic matching-based approaches necessitate large-scale datasets for effective training, which are typically unavailable in practical scenarios, hindering their competitive performance. To address this challenge, we propose a novel architecture for knowledge graphs known as LP-BERT, which incorporates a language model. LP-BERT consists of two primary stages: multi-task pre-training and knowledge graph fine-tuning. During the pre-training phase, the model acquires relationship information from triples by predicting either entities or relations through three distinct tasks. In the fine-tuning phase, we introduce a batch-based triple-style negative sampling technique inspired by contrastive learning. This method significantly increases the proportion of negative sampling while maintaining a nearly unchanged training time. Furthermore, we propose a novel data augmentation approach that leverages the inverse relationship of triples to enhance both the performance and robustness of the model. To demonstrate the effectiveness of our proposed framework, we conduct extensive experiments on three widely used knowledge graph datasets: WN18RR, FB15k-237, and UMLS. The experimental results showcase the superiority of our methods, with LP-BERT achieving state-of-the-art performance on the WN18RR and FB15k-237 datasets.

This work is partially supported by the National Key Research and Development Program of China (2021ZD0112901). D. Li and B. Zhu contributed equally.
Authors’ addresses: D. Li, M. Yi, and Y. He, Tencent, Beijing, China, 100084; e-mails: [danielsl, mingyyi, yukaihe]@tencent.com; B. Zhu, K. Xu (Corresponding author), and H. Wang, National Key Lab of Parallel and Distributed Processing, National University of Defense Technology, Changsha, China, 410073; e-mails: [zhubog, kele.xu]@gmail.com, wlm_w@163.com; S. Yang, Bioinformatics Center of AMMS, Beijing, China, 100071; e-mail: yangsen@nudt.edu.cn.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
2375-4699/2023/11-ART250 $15.00
https://doi.org/10.1145/3627704

ACM Trans. Asian Low-Resour. Lang. Inf. Process., Vol. 22, No. 11, Article 250. Publication date: November 2023.
1 INTRODUCTION

The applications of knowledge graphs (KGs) are apparent in both the industrial and academic fields [19], including question answering, recommendation systems, and natural language processing [40]. These applications have attracted considerable interest in constructing large-scale KGs. However, despite the sustained efforts, many previous KGs have suffered from incompleteness [33], as it is challenging to store all facts at once. To address this issue of incompleteness, many link prediction approaches have been explored. The aim of these approaches is to discover unannotated relations between entities to complete KGs, which is a critical and challenging task due to its potential to boost downstream applications.

Link prediction for KGs is also known as KG completion. Previous link prediction methods can be classified into two main categories: translational-distance-based approaches and semantic-matching-based approaches [48]. Translational-distance-based models typically embed both entities and relations into vector space and exploit scoring functions (SFs) to measure the distance between them. Although the distance representation of entity relations can be very diverse, it is difficult to predict the entity information that did not appear in the training phase. As a promising alternative, semantic-matching-based approaches utilize semantic information of entities and relationships and are capable of embedding those unseen entities based on their text description. Furthermore, due to the highly complex structure of the model and the slow training speed, the proportion of negative sampling is much lower for the training purpose, which leads to insufficient learning of negative sample entity information in the entity library and severely constrains the performance of the model.

To address the aforementioned issues, particularly to alleviate the poor prediction performance of the unseen node of the translational distance model and insufficient training of the text matching model, this article proposes a novel pre-training framework for KGs, namely LP-BERT. Specifically, LP-BERT employs semantic matching representation, which leverages a multi-task pre-training strategy, including a masked language model (MLM) task for context learning, a masked entity model (MEM) task for entity semantic learning, and a masked relation model (MRM) task for relational semantic learning. With these pre-training tasks, LP-BERT can learn relational information and unstructured semantic knowledge of structured KGs. Moreover, to solve the problem of insufficient training induced by the low negative sampling ratio, a negative sampling in a batch inspired by contrastive learning is proposed, which significantly increases the proportion of negative sampling while ensuring that the training time remains unchanged. At the same time, a data augmentation method based on the inverse relationship of triples is proposed to increase sample diversity, which can further boost performance.

To demonstrate the effectiveness and robustness of the proposed solution, we comprehensively evaluate the performance of LP-BERT on WN18RR, FB15k-237, and UMLS datasets. Without any additional modifications, LP-BERT outperforms a group of competitive methods [14, 30, 38, 46, 65].
and achieves state-of-the-art results on the WN18RR and UMLS datasets. The Hits@10 indicator is improved by 5% from the previous state-of-the-art result on the WN18RR dataset, while reaching 100% on the UMLS dataset.

The structure of the remainder of this article is as follows. Section 2 discusses the relationship between the proposed method and prior works. In Section 3, we provide the details of our methodology, while Section 4 describes comprehensive experimental results. Section 5 provides the conclusion of this article.

2 RELATED WORK

2.1 Knowledge Graph Embedding

KG embedding is a well-studied area, and comprehensive surveys can be found in [15, 48, 59]. Traditional methods could only utilize the structural information observed in triples to complete the KG. For example, TransE [5] and TransH [50] are two representative works that iteratively updated the vector representation of entities by calculating the distance between them. Convolutional Neural Networks (CNNs) have also demonstrated satisfactory performance in KG embedding [21, 29, 57]. Moreover, different types of external information, such as entity types, logical rules, and text descriptions, have been introduced to enhance the results [48]. For text descriptions, [37] first represented entities by averaging word embeddings contained in their names, where the word embeddings are learned from external corpora. [48] embedded entities and words into the same vector space by aligning the Wikipedia anchor points and entity names. [51] proposed Semantic Space Projection (SSP) to learn topics and KG embedding together by depicting the strong correlation between fact triples and text descriptions. KG embedding models [17] proposed semantic-driven versions for the three main loss functions for link prediction to treat the scores of negative triples differently by injecting background knowledge about relation domains and ranges into the loss functions. [54] proposed a simple yet efficient contrastive learning framework for tensor decomposition-based (TDB) KG embedding, which can shorten the semantic distance of the related entities and entity-relation couples in different triples and thus improve the performance of KG embedding. [20] proposed an INductive knowledge GRAph eMbedding method, InGram, that can generate embeddings of new relations as well as new entities at inference time. Given a KG, we define a relation graph as a weighted graph consisting of relations and the affinity weights between them. [62] proposed Modality-Aware Negative Sampling (MANS) for multi-modal knowledge graph embedding (MMKGE) to address the mentioned problems.

Despite the satisfactory performance of existing models, they typically learn the same text representation of entities and relationships, without considering the potential differences in meanings or importance of weights of words in different triples. To address these issues, Xu et al. [53] used a Long Short-Term Memory (LSTM) encoder with attention mechanism to construct context text representations for different relationships. An et al. [1] proposed an accurate text-enhanced KG embedding method that employs mutual attention between triple-specific relation mention and entity description.

Although these methods are effective in dealing with semantic changes of entities and relations across different triples, they do not fully leverage syntactic and semantic information in large-scale free text data as they only use entity descriptions, relation mentions, and word co-occurrence with entities. Recently, pre-training paradigms such as KG-BERT [56], MLMLM [9], and StAR [46] have been introduced. However, due to the high complexity and slow training speed of these models, the proportion of negative sampling is typically far lower than that of previous works, resulting in insufficient learning of negative sample entity information in the entity library.

In comparison to the aforementioned methods, our approach introduces a contrastive learning strategy. During training, a novel negative sampling approach is implemented in the batch, which
exponentially increases the proportion of negative sampling and helps to alleviate the issue of insufficient learning. Additionally, we optimize the pre-training strategy to enable the model to learn not only context knowledge but also the element information of the left entity, relationship, and right entity, leading to significantly improved performance of the model.

2.2 Link Prediction

Link prediction has been a popular research area in KG embedding in recent years. Various approaches have been proposed to address this task. KBGAT [28] proposed an attention-based feature embedding method to capture entity and relation features in the entity’s neighborhood. AutoSF [63] aimed to automatically design SFs for distinct KGs using AutoML techniques. CompGCN [44] developed a novel graph convolutional framework that jointly embeds nodes and relations in a relational graph using a variety of entity-relation composition operations from KG embedding techniques. Meta-KGR [24] was a meta-based multi-hop reasoning method that uses meta-learning to learn effective meta-parameters from high-frequency relations that could quickly adapt to few-shot relations. ComplEx-N3-RP [8] proposed a self-supervised training objective for multi-relational graph representation learning by incorporating relation prediction into the commonly used 1vsAll objective. AttH [7] was a class of hyperbolic KG embedding models that capture hierarchical and logical patterns by combining hyperbolic reflections and rotations with attention. RelAtt [60] leverages the relation and neighborhood information to compute the importance of neighbors by attention mechanism.

RotatE [39] defined each relation as a rotation from the source entity to the target entity in the complex vector space, enabling it to model and infer various relation patterns such as symmetry/antisymmetry, inversion, and composition. HAKE [65] combined TransE and RotatE to model entities at different levels of a hierarchy while distinguishing entities at the same level. GAAT [49] integrated an attenuated attention mechanism and assigned diverse weights to relation paths to acquire information from neighborhoods. STAR [46] partitioned a triple into two asymmetric parts as in the translation distance-based graph embedding approach and encoded both parts into contextualized representations by a Siamese-style textual encoder. [16] introduced a simple theoretical framework for rank-based metrics upon which they investigated two avenues for improvements to existing metrics via alternative aggregation functions and concepted from probability theory. [66] built upon representative GCN-based KGC models and introduced variants to find which factor of GCNs is critical in KGC. [42] introduced the Multi-partition Embedding Interaction Improved beyond block term format (MEIM) model, with independent core tensor for ensemble effects and soft orthogonality for max-rank mapping. However, the pre-training stage of textual models could only learn the context knowledge of the text, while ignoring the graph structure.

2.3 Pre-training of Language Model

Link prediction is an active research area in KG embedding and has received a lot of attention in recent years [7, 8, 46, 63, 65]. The pre-training language model method can be divided into two categories: feature-based method and fine-tuning based method [11, 32]. Traditional word embedding methods, such as Word2Vec [25] and Glove [31], aimed to use feature-based methods to learn context semantic information and express words in the form of feature vectors. ELMo [34] extends traditional word embedding to context-aware word embedding where polysemy can be properly handled. Different from feature-based approaches, the MLM-based pre-training method used in BERT [13] opened up a pre-training paradigm of a language model based on transformer structure. RoBERTa [22] carefully measures the impact of key hyper-parameters and training data size and further enhances the effect. SpanBERT [18] extended BERT by masking contiguous random spans rather than random tokens and training the span boundary representations to predict the
entire content of the masked span without relying on the individual token representations within it. MacBERT [10] improves upon RoBERTa in several ways, especially the masking strategy that adopts MLM as a correction.

Recently, the language model of pre-training has also been explored in the context of KG [6, 47, 56]. Wang et al. [47] learned context embeddings on entity-relational chains (sentences) generated by random walk in KG and initialized them as KG embedding models such as TransE [5]. COMET [6] used GPT to generate a given head phrase and a tail phrase tag of relation type in the knowledge base, which is not completely suitable for comparing the patterns of two entities with known relations. KG-BERT [56] used the MLM method for pre-training on the triple data to learn the corpus information of the KG scene. GenKGC [52] converts KG completion to a sequence-to-sequence generation task with the pre-trained language model. PALT [36] presents a parameter-light transfer learning approach of pretrained language models for KG completion that only tunes a few new parameters while keeping the original language model parameters fixed instead of fine-tuning by modifies all language model parameters. KGLM [58] introduces a new entity/relation embedding layer that learns to differentiate distinctive entity and relation types, therefore allowing the model to learn the structure of the KG. Unlike these studies, we use a multi-task pre-training strategy based on MLM, MEM, and MRM so that the model can learn not only context corpus information but also the association information of triples at the semantic level.

3 METHODOLOGY

3.1 Task Formulation

A KG, denoted as \( G \), can be represented as a set of triples \( G = \{E_h, R, E_t\}, i \in [1, N] \), where \( N \) is the number of triples in \( G \). In this representation, \( E_h \) and \( E_t \) represent the head entity and tail entity, respectively, and there is a directed edge with attribute \( R \) from \( E_h \) to \( E_t \). It is important to note that there could exist multiple edges from \( E_h \) to \( E_t \). All entities and relations are typically succinct textual units composed of several tokens. Additionally, each entity is associated with a description, denoted as \( D_h \) and \( D_t \), which offers an elaborate explication of the entity. The task of link prediction aims to determine the existence of a specific relation between two entities. More specifically, given \( E_h \) and \( E_t \) (along with their descriptions \( D_h \) and \( D_t \) ), the objective is to predict whether the triple \( \{E_h, R, E_t\} \) holds true within the context of the KG \( G \).

3.2 Overall Framework

Figure 1 illustrates the overall framework of our proposed LP-BERT. It includes two procedures for link prediction: multi-task pre-training and knowledge fine-tuning. During the pre-training stage, in addition to the widely used MLM task [13], we propose two novel pre-training tasks: the MEM task and the MRM task as demonstrated in Figure 2. By utilizing these three pre-training tasks in parallel, LP-BERT can learn both the context information of the corpus and the semantic information of the head-relation-tail triples.

During the fine-tuning stage, we adopt a triple-style negative sampling method in a batch, which is inspired by contrastive learning. This method significantly increases the proportion of negative sampling while keeping the training time almost unchanged. Additionally, we propose a data augmentation method based on the inverse relationship of triples to enhance sample diversity.

3.3 Multi-task Pre-training

We propose to pre-train the LP-BERT model with three tasks. Although they require different masking strategies, they also share the same input, which concatenates entities, relation, and entity
Multi-task Pre-training

- **Masked Language Model (MLM)**: Random tokens are masked.
- **Masked Entity Model (MEM)**: Either the head entity or the tail entity is masked.
- **Masked Relation Model (MRM)**: The relation is masked.

**Fig. 2.** An instance demonstrating the pre-training tasks. The entities, relations, and entity descriptions are concatenated to form a sequence. In the Mask Language Model (MLM) task, random tokens are masked. In the Mask Entity Model (MEM) task, either the head entity or the tail entity is masked, denoted by subscript “h” or “t,” respectively. Finally, in the Mask Relation Model (MRM) task, the relation is masked. It is important to note that these pre-training tasks can be combined using the multi-task learning paradigm during the pre-training phase.

### 3.3.1 Mask Entity Modeling

In the MEM task, the input entity sequence is masked and the model is tasked with recovering the masked entity based on another entity and relation. To prevent information leakage, the corresponding entity description is also masked. Since each triple includes descriptions in a triple as a whole sequence:

$$\hat{X} = [CLS]||E_h||D_h||[SEP]||R||[SEP]||E_t||D_t||[SEP].$$  

(1)

Here, symbol || represents sentence concatenation, and “[CLS]” and “[SEP]” are two reserved tokens in BERT [13], denoting the beginning and separation of the sequence. More details of the three pre-training tasks are described in the following sections.
two entities, MEM can mask either the head entity or the tail entity, but not both simultaneously. As an example, consider MEM on the tail entity. The input is as follows:

$$\tilde{X} = [CLS][E_h][D_h][SEP][R][MASK][PAD][SEP].$$  

(2)

Here, we use “[MASK]” to represent the masked tokens. As the tail entity is masked in the MEM task, its corresponding description is replaced with a reserved token “[PAD]” to prevent the model from inferring the entity solely from its description. It is worth noting that both “[MASK]” and “[PAD]” could contain multiple tokens since they share the same length with the tail entity and its description. The objective of the prediction is to recover the tail entity without considering its description. Similarly, for predicting the head entity, LP-BERT randomly masks the head entity and predicts the corresponding tokens. In the pre-training process, both head and tail entities are masked with the same probability.

To predict the tokens in the entity, a classifier that combines a multi-layer perceptron (MLP) and Batch-Norm layer is built on top of the LP-BERT encoder. This classifier outputs a probability matrix of the prediction results:

$$p = \text{BN} \circ \text{GeLU} \circ \text{MLP} \circ \text{LP-BERT}(\tilde{X}),$$  

(3)

in which \(\circ\) denotes function composition. Each token has a probability vector of the vocabulary size, but the prediction results are not involved in the loss calculation except tokens of the masked entity.

3.3.2 Mask Relation Modeling. A similar sample construction strategy to MEM is used for the MRM task (as shown in Algorithm 1). Instead of masking one of the two entities in the triple, MRM replaces tokens in the relation with “[MASK]” while preserving the head and tail entities and descriptions. MRM then drives the model to predict the corresponding relation between the two entities. The masked sample can be represented as follows:

$$\tilde{X} = [CLS][E_h][D_h][SEP][R][MASK][PAD][SEP].$$  

(4)

3.3.3 Mask Language Modeling. In order to coexist with MEM and MRM, unlike BERT, which employs [13] random masking prediction of all tokens in the sequence, the proposed MLM method only makes local random masking for the specific text range of the sample. The random masking strategy is as follows:

- For the head entity prediction task in MEM, random mask only in token sequences of \(E_t\) and \(D_t\).
- For the tail entity prediction task in MEM, random mask only in token sequences of \(E_h\) and \(D_h\).
- For the MRM task, random mask only in token sequences of \(E_h, D_h, E_t,\) and \(D_t\).

In this way, MLM drives the model to learn the corpus’s context information. More important, though MRM and MEM are exclusive, they are both compatible with MLM. Therefore, MLM is conducted together with either MEM or MRM during the pre-training procedure for a mini-batch. At the same time, doing masking is equivalent to doing dropout-like regularization for the text features of MEM and MRM tasks, and it can improve the performance of MEM and MRM, as shown in the experiments.

Algorithm 1 shows more details about the construction of samples for pre-training LP-BERT. Lines 20–25 and 26–31 show the procedure of MEM for the head entity and tail entity, respectively, with MLM. Lines 32–36 show the procedure of MRM with MLM. Lines 1–15 of Algorithm 1 display in detail the strategy for masking tokens, i.e., MLM.
**ALGORITHM 1:** Sample Construction in the Pre-training Phase

**Require:** Tokens: Token list of Triples

**Ensure:** $x$: masked input, $y_1$: MRM or MEM target, $y_2$: MLM target

1: function MLM($x$, $y_2$, region)
2:     for $i$ in region do
3:         if random[0, 1]<0.15 then
4:             $y_2[i] = x[i]$
5:         end if
6:     if random[0, 1]<0.8 then
7:         $x[i] = [\text{MASK}]$
8:     else
9:         if random[0, 1]>0.5 then
10:            $x[i] = \text{sampe(Vocab)}$
11:        end if
12:     end if
13: end for
14: return $x$, $y_2$
15: end function

16: $x \leftarrow \tilde{X}$ in Equation (1)
18: $y_1, y_2 \leftarrow [\text{PAD}] \ast \text{len(}\tilde{X})$
19: random_state $\leftarrow \text{random([0, 1])}$
20: if random_state$<0.4$ then
21:     $y_1[E_h] \leftarrow x[E_h]$
22:     $x[E_h] \leftarrow [\text{MASK}] \ast \text{len}(E_h)$
23:     $x[D_h] \leftarrow [\text{PAD}] \ast \text{len}(E_h)$
24:     $x, y_2 \leftarrow \text{MLM}(x, y_2, R \parallel E_t \parallel D_t)$
25: end if
26: if 0.4$\leq$random_state$<0.8$ then
27:     $y_1[E_t] \leftarrow x[E_t]$
28:     $x[E_t] \leftarrow [\text{MASK}] \ast \text{len}(E_t)$
29:     $x[D_t] \leftarrow [\text{PAD}] \ast \text{len}(E_t)$
30:     $x, y_2 \leftarrow \text{MLM}(x, y_2, E_h \parallel D_h \parallel R)$
31: end if
32: if random_state$\geq0.8$ then
33:     $y_1[R] \leftarrow x[R]$
34:     $x[R] \leftarrow [\text{MASK}] \ast \text{len}(R)$
35:     $x, y_2 \leftarrow \text{MLM}(x, y_2, \text{regions of } x \text{ other than } R)$
36: end if
37: return $x, y_1, y_2$

### 3.3.4 Pre-training Loss Designing

Since the strategies for constructing samples in MEM and MRM tasks are mutually exclusive, the prediction of the head entity and tail entity cannot be simultaneously predicted for the triple samples trained by the same input model. To ensure the generalization ability of the model, we use the MIM task as a unified expression of MEM and MRM tasks and define the loss function as follows:

$$L = L_{MLM}(y', y) + L_{MIM}(y', y|\alpha),$$

(5)

where $L$ is the final loss, $y'$ and $y$ are the prediction objectives and the gold label, respectively, and $\alpha$ is the random number uniformly distributed in the interval $[0, 1]$. Details of $L_{MIM}$ are shown as
Fig. 3. There are $n$ rows and $n$ columns in the batch, with each block containing one sample, so there are $n^2$ samples in the batch. The sample with a value of 1 above the diagonal is a positive sample in the batch, and the sample with a value of 0 in the $i_{th}$ row is the diagonal of the negative sampling sample for $E_h R_i$. For $E_h R_i$ and $E_t j$ in the $i_{th}(1 \leq i \leq n)$ element of the batch, they can only generate one positive sample and $n-1$ negative samples. Therefore, there are $n^2$ samples in the batch, including $n$ positive samples and $n(n-1)$ negative samples.

follows:

$$L_{MIM}(y', y|\alpha) = \begin{cases} 
L_{MEM}(y', y) & 0.0 \leq \alpha < 0.4 \\
L_{MEM}(y', y) & 0.4 \leq \alpha < 0.8 \\
L_{MRM}(y', y) & 0.8 \leq \alpha < 1.0 
\end{cases} \quad (6)$$

3.4 Fine-tuning for Knowledge Graph

3.4.1 Knowledge Representation. The sample construction method in the fine-tuning stage differs from that in the pre-training stage. Inspired by STAR [46], for each triple, we concatenate the $E_h$ and $R$ texts. Then, the pre-trained LP-BERT model uses the Siamese structure [27] to encode $E_h | R$ and $E_t$ separately. The fine-tuning objective of the model is to minimize the distance between the two representations of the positive sample and maximize the distance between the negative ones. Here, a positive sample is a $(E_h, R, E_t)$ triple that exists in the knowledge base, while a negative sample is one that does not.

Since a triple $(E_h, R, E_t)$ is split into $E_h | R$ and $E_t$, the KG can only provide positive samples. Therefore, to perform binary classification during fine-tuning, we propose a simple but effective method for generating negative samples. As illustrated in Figure 3, for a mini-batch of size $n$, we interleave $E_h | R_i$ and $E_t j (1 \leq i, j \leq n)$ to take $E_h | R_i, E_t j (i \neq j)$ as negative samples. Therefore, for a mini-batch, LP-BERT forwards only twice for $n^2$ sample distances, greatly increasing the proportion of negative sampling and reducing the training time. The detailed procedure of constructing negative samples for fine-tuning is shown in Algorithm 2.

3.4.2 Triple Augmentation. The above pair-based KG representation method has limitations because it cannot represent $(E_h, R, E_t)$ pairs directly in the head entity prediction task. Specifically, when constructing negative samples, we can only perform negative sampling for $E_t$ and not for $E_h$, which limits the diversity of negative samples, especially when there are few relationships in the dataset. To address this limitation, we propose a dual relationship data enhancement method. For each relationship $R$, we define a corresponding inverse relationship $R_{rev}$. For the head entity prediction sample in the form of $(?, R, E_t)$, we rewrite it in the form of $(E_t, R_{rev}, ?)$ to enhance the data.

In this way, we can use the vector representation of $(E_t, R_{rev})$ and $E_h$ to replace $(E_h, R, E_t)$, thereby improving the diversity of sampling and the robustness of the model. Additionally, we use the mixed-precision strategy of fp16 and fp32 to reduce the GPU memory usage of gradient calculation, which enables larger values of $n$ and an increased negative sampling ratio.
**Algorithm 2:** Batch Sample Construction in the Fine-tuning Phase

**Require:** \(x_1: E_{hR}\) batch tokens, \(x_2: E_t\) batch tokens, dict: a dict can get positive Entities for each \(E_h\)\|\(R\)

**Ensure:** \(p\): model predict results, \(y\): ground truth

1: \(\text{Emb}_1 \leftarrow \text{Encoding of } x_1\)
2: \(\text{Emb}_2 \leftarrow \text{Encoding of } x_2\)
3: \(p \leftarrow \text{cosine similarity of } \text{Emb}_1 \text{ and } \text{Emb}_2\)
4: \(y = []\)
5: for \(E_{hR} \in x_1\) do
6:     for \(E_t \in x_2\) do
7:         if \(E_t \in \text{dict}[E_{hR}]\) then
8:             \(y.append(1)\)
9:         else
10:             \(y.append(0)\)
11:     end if
12: end for
13: end for
14: return \(p, y\)

### 3.4.3 Fine-tuning Loss Designing

We designed two distance calculation methods to jointly calculate the loss function:

\[
L = L_1(V_{E_{hR}}, V_{E_t}) + L_2(V_{E_{hR}}, V_{E_t}),
\]

where \(V_{E_{hR}}\) and \(V_{E_t}\) are encoded vectors of \(E_{hR}\) and \(E_t\), respectively.

\[
L_1(d_1) = \begin{cases} 
\alpha_t(1 - d_1)^{\gamma}\log(d_1) & y = 1 \\
\alpha_t(d_1)^{\gamma}\log(1 - d_1) & y \neq 1 
\end{cases}
\]

\[
L_2(d_2) = \begin{cases} 
\text{Sigmoid}(\text{sum}(d_2)) & y = 1 \\
1 - \text{Sigmoid}(\text{sum}(d_2)) & y \neq 1 
\end{cases}
\]

where

\[
\begin{align*}
d_1 &= \frac{V_{E_{hR}}V_{E_t}}{\|V_{E_{hR}}\|\|V_{E_t}\|}, \\
d_2 &= \|V_{E_{hR}} - V_{E_t}\|
\end{align*}
\]

where \(\alpha\) is used to adjust the weights of positive and negative samples, and \(\gamma\) is used to adjust the weights of samples that are difficult to distinguish. We utilize two different dimensional distance calculation methods to determine the distance relationship between the vector pairs in multi-task learning.

### 4 EXPERIMENTS

In this section, we first provide details about the experimental settings. Then, we demonstrate the effectiveness of our proposed model on multiple widely used datasets, including WN18RR [12], FB15k-237 [41], and UMLS [12] benchmark datasets. We also conduct ablation studies to tease apart the contribution of each improvement in our standard model while keeping other structures as they are. Finally, we perform extensive analysis of the prediction performance for unseen entities, and a case study is provided to show the effectiveness of our proposed LP-BERT.

#### 4.1 Experiment Settings and Datasets

We evaluated the proposed models on three benchmark datasets: WN18RR [12], FB15k-237 [41], and UMLS [12]. WN18RR is a dataset adopted from WordNet [26] for the link prediction task, consisting of English phrases and their corresponding semantic relations. FB15k-237 is a subset of
Freebase [4], a dataset of real-world named entities and their relations. Both WN18RR and FB15k-237 are updated versions of WN18 and FB15k [5], respectively, with inverse relations and data leakage removed, making them popular benchmarks for KG completion. UMLS is a small KG containing medical semantic entities and their relations. The summary statistics of the datasets are presented in Table 1. We provide details of the experimental settings and demonstrate the effectiveness of our proposed model on these datasets. Additionally, we conducted ablation studies to justify the contribution of each improvement, and extensive analysis of the prediction performance for unseen entities. Finally, we present a case study to showcase the effectiveness of our proposed LP-BERT.

We implemented LP-BERT using the PyTorch framework on a workstation with an Intel Xeon processor, 64GB of RAM, and a Nvidia P40 GPU for training purposes. We used the AdamW optimizer with 5% warmup steps. For the hyperparameters in LP-BERT, we set the number of epochs to 50, the batch size to 32, and the learning rate to $10^{-4}/5 	imes 10^{-5}$ for the linear and attention parts initialized. The early-stop epoch number was set to three. In the KG fine-tuning phase, we set the batch size to 64 for WN18RR, 120 for FB15k-237, and 128 for UMLS based on the best Hits@10 on the development dataset. The learning rate was set to $10^{-3}/5 	imes 10^{-5}$ for the linear and attention parts initialized with LB-BERT. The number of training epochs was seven for WN18RR and FB15k-237, and 30 for the UMLS dataset. We set $\alpha = 0.8$ on WN18RR and UMLS, $\alpha = 0.5$ on FB15k-237, and $\gamma = 2$ in Equation (8). The batch size was set to 64, and the learning rate was initialized as 0.00001. The hyper-parameters are tuned during training to optimize the model’s performance on the validation set. To ensure a fair evaluation, we randomly select a portion of the samples from the training set to create a validation set. The intention is to have the number of samples in the validation set approximate that of the test set. It is important to note that the parameters obtained through this tuning process represent the optimal parameters for the validation set, rather than for the test set. The purpose of the validation set is to assess and fine-tune the model’s performance, while the test set serves as an independent benchmark to evaluate the model’s generalization ability. In summary, the parameters are adjusted based on their performance on the validation set, and these optimized parameters reflect the model’s optimal performance on the validation set, not the test set.

Training for the model took approximately 20 hours, with pre-training comprising around 16 hours and fine-tuning taking approximately 4 hours. This training was conducted on the WN18RR dataset. During the inference phase, all entities, other than the correct ones, were considered as incorrect candidates. This approach allowed for the identification of potential issues with the head or tail entities. The trained model utilized the “filtered” settings to improve the ranking of corrupt triples. The evaluation metrics encompassed two aspects. First, Hits@N measured the proportion of test instances that included the correct candidates within the top $N$ predictions. Second, the average rank (MR) and the average reciprocal rank (MRR) provided insights into the absolute ranking of the predictions.

| Entities | WN18RR | FB15k-237 | UMLS |
|----------|--------|-----------|------|
| Relations | 11 | 237 | 46 |
| Train samples | 86,835 | 272,115 | 5,216 |
| Valid samples | 3,034 | 17,535 | 652 |
| Test samples | 3,034 | 20,466 | 661 |

Table 1. The Summary Statistics of the Used Datasets, Including WN18RR, FB15k-237, and UMLS
Table 2. Experimental Results on WN18RR, FB15k-237, and UMLS Datasets

| Methods          | WN18RR | FB15k-237 | UMLS  |
|------------------|--------|-----------|-------|
|                  | Hits@1↑| Hits@3↑   | Hits@10↑ | MR↑ | MRR↑ | Hits@1↑ | Hits@3↑ | Hits@10↑ | MR↑ | MRR↑ | Hits@10↑ | MR↓ |
| TransE [5]       | 0.043  | 0.441     | 0.532  | 2300 | 0.243 | 0.198  | 0.376  | 0.441  | 323 | 0.279 | 0.989 | 1.84 |
| DistMult [55]    | 0.412  | 0.470     | 0.504  | 7000 | 0.444 | 0.199  | 0.301  | 0.446  | 512 | 0.281 | 0.846 | 5.52 |
| ComplEx [43]     | 0.409  | 0.469     | 0.530  | 7882 | 0.449 | 0.194  | 0.297  | 0.450  | 546 | 0.278 | 0.967 | 2.59 |
| R-GCN [35]       | 0.080  | 0.137     | 0.207  | 6700 | 0.123 | 0.100  | 0.181  | 0.300  | 600 | 0.164 |    -   |    - |
| ConvE [12]       | 0.419  | 0.470     | 0.531  | 4464 | 0.456 | 0.225  | 0.341  | 0.497  | 245 | 0.312 | 0.990 | 1.51 |
| KBAT [28]        | -      | -         | 0.554  | 1921 | 0.412 | -      | -      | 0.331  | 270 | 0.157 |    -   |    - |
| QuatE [61]       | 0.436  | 0.500     | 0.564  | 3472 | 0.481 | 0.221  | 0.342  | 0.495  | 176 | 0.311 |    -   |    - |
| RotatE [39]      | 0.428  | 0.492     | 0.571  | 3340 | 0.476 | 0.241  | 0.375  | 0.533  | 177 | 0.338 |    -   |    - |
| TuckER [3]       | 0.443  | 0.482     | 0.526  | -    | 0.470 | 0.266  | 0.394  | 0.544  | -   | 0.358 |    -   |    - |
| AttH [7]         | 0.443  | 0.499     | 0.573  | -    | 0.486 | 0.252  | 0.384  | 0.540  | -   | 0.348 |    -   |    - |
| ConvH [2]        | 0.453  | 0.515     | 0.579  | -    | 0.496 | 0.247  | 0.381  | 0.54  | -   | 0.345 |    -   |    - |
| Dense [23]       | 0.443  | 0.508     | 0.579  | 3052 | 0.491 | 0.256  | 0.384  | 0.535  | 169 | 0.349 |    -   |    - |
| Rot-Pro [38]     | 0.397  | 0.482     | 0.577  | -    | 0.457 | 0.246  | 0.383  | 0.540  | -   | 0.344 |    -   |    - |
| QuatDE [14]      | 0.438  | 0.509     | 0.586  | 1977 | 0.489 | 0.268  | 0.400  | 0.563  | 90  | 0.365 |    -   |    - |
| LineaRE [30]     | 0.453  | 0.509     | 0.578  | 1644 | 0.495 | 0.264  | 0.391  | 0.545  | 155 | 0.357 |    -   |    - |
| CapsE [45]       | -      | -         | 0.559  | 718  | 0.415 | -      | -      | 0.356  | 403 | 0.150 |    -   |    - |
| RESCAL-DURA [64] | 0.455  | -         | 0.577  | -    | 0.498 | 0.276  | -      | 0.550  | -   | 0.368 |    -   |    - |
| HAKE [65]        | 0.452  | 0.516     | 0.582  | -    | 0.497 | 0.250  | 0.381  | -      | -   | 0.346 |    -   |    - |

Semantic matching models

| Methods          | Parameters          | WBERT [56] | StAR [46] | GenKGC [52] | PALT [36] | KGLM [58] | LP-BERT |
|------------------|---------------------|------------|-----------|-------------|-----------|-----------|---------|
|                  |                     | 0.041      | 0.302     | 0.524       | 0.512     | 0.572     | 0.343   |
|                  |                     | 0.243      | 0.491     | 0.709       | 0.51      | 0.581     | 0.563   |
|                  |                     | 0.287      | 0.403     | 0.535       | -         | 0.583     | 0.752   |
|                  |                     | 0.303      | 0.538     | 0.741       | 0.40      | 0.606     | 0.462   |
|                  |                     | 0.330      | 0.538     | 0.741       | 0.40      | 0.606     | 0.462   |
|                  |                     | 0.343      | 0.563     | 0.752       | 0.92      | 0.482     | 0.223   |

The underlined numbers indicate the optimal results and the bold numbers indicate the sub-optimal results. It can be observed that the proposed L-BERT model achieves state-of-the-art performance on multiple evaluation metrics on both the WN18RR and UMLS datasets. Moreover, it outperforms other semantic matching models on the FB15k-237 dataset. In the table, the symbol ↑ signifies that higher values indicate better performance, while ↓ signifies that lower values indicate better performance.

4.2 Results

We evaluated the proposed methods and competitive approaches for link prediction tasks, including translational-distance-based approaches and semantic-matching-based methods. In our experiments, we tested 18 widely used translational distance models, such as TransE [5], DistMult [55], ComplEx [43], R-GCN [35], ConvE [12], KBAT [28], QuatE [61], RotatE [39], TuckER [3], AttH [7], DensE [23], Rot-Pro [38], QuatDE [14], LineaRE [30], CapsE [45], RESCAL-DURA [64], and HAKE [65]. We also included semantic-matching-based methods, KG-BERT [56] and StAR [46], in our experiments.

Table 2 presents the detailed results of our experiments. As shown in the table, L-BERT achieved state-of-the-art or competitive performance on all three widely used datasets, WN18RR, FB15k-237, and UMLS. The improvement is particularly significant in terms of Hits@10 and Hits@3 due to the superior generalization performance of the multi-task pre-training textual encoding approach. We further analyze this in the section below. Additionally, L-BERT outperformed all other methods by a large margin in terms of Hits@3, Hits@10 on WN18RR, and Hits@10 on UMLS. Although it only achieved inferior performance on the FB15k-237 dataset and Hits@1 and MRR of the WN18RR dataset compared to translational distance models, it still remarkably outperformed other semantic matching models such as KG-BERT and StAR from the same genre.
by introducing structured knowledge. Notably, LB-BERT outperformed STAR [46], the previous state-of-the-art model, on all three datasets with only one-third the number of parameters. For the WN18RR dataset, Hits@1 increased from 0.243 to 0.343, and Hits@3 increased from 0.491 to 0.563.

The experimental results indicate that semantic matching models perform well in topK recall evaluation methods, but their Hits@1 result is significantly inferior to that of translation distance models. This is because the features of semantic matching models are based on text, which leads to the vector representation of similar entities in the text being close and difficult to distinguish. Although translation distance models perform well on Hits@1, they are incapable of understanding text semantics. For new entities not seen in the training set, the prediction results of translation distance models are random, while those of semantic matching models are reliable. This is why Hits@3 and Hits@10 of LP-BERT can far exceed those of translation distance models, enabling them to achieve state-of-the-art performance.

Similar to KG-BERT and StAR, our model relied on BERT and we compare LP-BERT with KG-BERT and StAR on WN18RR in detail, including different initialization manners. As shown in Table 3, LP-BERT consistently achieves superior performance over most metrics. The evaluation effect of the LB-BERT model based on RoBERTa-base has exceeded the evaluation effect of KG-BERT and StAR based on RoBERTa-large. As for empirical efficiency, due to the small number of model parameters and the strategy of negative sampling based on the batch in the training process, our model is faster than KG-BERT and StAR for both the training and inference phases.

4.3 Ablation Study

In this part, we tease apart each module from our standard model and keep other structures as they are, with the goal to justify the contribution of each improvement. The ablation experiments are conducted on the WN18RR dataset, and we find similar performance on all three datasets. Table 4 shows the results.

For the Batch-Sampling component, increasing the number of negative samples is beneficial to the training process. However, due to the high computational overhead, the negative sampling multiplier of the previous work related to language-based models is very low. The Batch-Sampling method significantly reduces the computational overhead and increases the sampling multiplier by a factor of Batch Size with the same computational overhead. After adding a batch-based triple-style negative sampling strategy combined with focal loss, the Hits@1 and MRR significantly improve by 0.14% and 0.167%, respectively, as in the second line. For the MLM/MEM/MRM component, MLM is the traditional context-based corpus pre-training method, and adding this

Table 3. Quantitative Comparisons with KG-BERT and StAR on WN18RR Datasets

| Weight Initialization | Hits@1↑ | Hits@3↑ | Hits@10↑ | MR↓ | MRR↑ | Train↓ | Inference↓ |
|-----------------------|---------|---------|----------|-----|------|--------|-----------|
| KG-BERT               | RoBERTa-base | 0.130 | 0.320 | 0.636 | 84  | 0.278 | 4h  | 32h |
| StAR                  | RoBERTa-base | 0.202 | 0.410 | 0.621 | 71  | 0.343 | 2h  | 0.9h |
| **LP-BERT**           | RoBERTa-base | 0.278 | 0.502 | 0.708 | 79  | 0.424 | 0.8h | 0.8h |
| KG-BERT               | RoBERTa-large | 0.119 | 0.387 | 0.698 | 95  | 0.297 | 7.9h | 92h |
| StAR                  | RoBERTa-large | 0.243 | 0.491 | 0.709 | 51  | 0.401 | 5.5h | 1h  |
| **LP-BERT**           | RoBERTa-large | 0.306 | 0.517 | 0.718 | 69  | 0.444 | 2.2h | 1h  |
| **LP-BERT**           | LP-BERT-base | 0.343 | 0.563 | 0.752 | 92  | 0.482 | 0.8h | 0.8h |

"Train" denotes the time for per training epoch, and "Inference" denotes total inference time on test set. The values were collected using Tesla P40 without mixed precision. The symbol ↓ signifies that lower values correspond to enhanced performance, while ↑ indicates that higher values result in improved performance.

"Train" denotes the time for per training epoch, and "Inference" denotes total inference time on test set. The values were collected using Tesla P40 without mixed precision. The symbol ↓ signifies that lower values correspond to enhanced performance, while ↑ indicates that higher values result in improved performance.
Table 4. Ablation Study for LP-BERT on the WN18RR Dataset

| MLM  | MEM | MRM | Batch-Sampling | Hits@1↑ | Hits@3↑ | Hits@10↑ | MRR↑ | MRR↓ |
|------|-----|-----|----------------|--------|--------|---------|------|------|
| ✓    | ✓   | ✓   | ✓              | 0.136  | 0.306  | 0.499   | 143  | 0.257|
| ✓    | ✓   | ✓   | ✓              | 0.278  | 0.502  | 0.708   | 79   | 0.424|
| ✓    | ✓   | ✓   | ✓              | 0.307  | 0.530  | 0.721   | 101  | 0.449|
| ✓    | ✓   | ✓   | ✓              | 0.300  | 0.540  | 0.718   | 103  | 0.447|
| ✓    | ✓   | ✓   | ✓              | 0.329  | 0.555  | 0.733   | 109  | 0.469|
| ✓    | ✓   | ✓   | ✓              | 0.343  | 0.563  | 0.752   | 92   | 0.482|

The symbol ✓ represents the approach accompanied by its corresponding operations, whereas × indicates the approach without said strategy. The symbol ↓ signifies that lower values correspond to enhanced performance, while ↑ indicates that higher values result in improved performance.

Table 5. Performance of LP-BERT on Unseen Entities

| Models  | Hits@1↑ | Hits@3↑ | Hits@10↑ | MRR↑ | MRR↓ |
|---------|---------|---------|----------|------|------|
| TransE  | 0.0010  | 0.0010  | 0.0010   | 0.0010 | 21708|
| DistMult| 0.0000  | 0.0000  | 0.0000   | 0.0000 | 33955|
| ComplEx | 0.0000  | 0.0000  | 0.0000   | 0.0000 | 24678|
| RotatE  | 0.0010  | 0.0010  | 0.0010   | 0.0010 | 21023|
| LinearRE| 0.0010  | 0.0010  | 0.0010   | 0.0010 | 21502|
| QuatDE  | 0.0010  | 0.0010  | 0.0010   | 0.0010 | 21301|
| MLMLM   | 0.0490  | 0.0932  | 0.1413   | 0.0812 | 6324 |
| StAR    | 0.1108  | 0.2355  | 0.4053   | 0.2072 | 535  |
| LP-BERT | **0.1204**| **0.2978**| **0.4919**| **0.2434**| 1998 |

Pre-training allows the model to have better parameter initialization under the KG complementation task, so the prediction effect is improved compared with no pre-training. Meanwhile, the KG complementation task has obvious structured information compared with the natural language task, i.e., <entity, relation, entity> structure. However, the MLM strategy does not fully excavate the relationship information of triplets. In order to make the model understand this structured information better, we add two tasks of MEM and MRM to the pre-training. The multi-task pre-training strategy combined with MLM, MEM, and MRM makes the model evaluation result optimal.

### 4.4 Unseen Entities

To verify the prediction performance of LP-BERT on unseen entities, we re-split the dataset. Specifically, we randomly select 10% of the entity triples as the validation set and test set, respectively, to ensure that the training set, validation set, and test set don’t overlap on any entity. We then re-train and evaluate LP-BERT as well as other baselines, of which the results are shown in Table 5.

As can be seen from the table, all models experienced a dramatic performance drop across the five metrics. In particular, the distance-based methods are inferior in coping with unseen entities. As mentioned above, such methods only encode the relationship and distance between entities without including semantic information, and thus are incapable of encoding entities not seen in the training set. Contrastively, pre-training-based models, including MLMLM, StAR, and our LP-BERT, have displayed their ability to cope with unseen entities. Furthermore, our LP-BERT surpasses MLMLM and StAR on almost all metrics, proving its superiority for processing unseen entities. Especially, LP-BERT outperforms StAR, the previous state of the art, by over 6 to 9 points on Hits@3 and Hits@10. However, the score of LP-BERT on the mean rank metric is not as good as other metrics, indicating LP-BERT performs worse on those failed entities.
4.5 Case Study

To demonstrate the performance of LP-BERT further, we conducted additional case studies on the WN18RR dataset and visualize the results in Table 6. Each row in the table denotes a real sample randomly selected from the test set. The first column represents a triple formatted as (left entity, relation, ?) ← right entity. The prediction models use the left entity and the relation to predict the right entity. From the second column to the fourth column, we present the Top 5 ranked entities with the highest predicted probability, which were obtained from different pre-training approaches (including the proposed LP-BERT-based pre-training, MLM-based pre-training, and without pre-training). The entities are ordered using the predicted probability. The correct answers are highlighted using boldface. The number in each element is the order of the correctly predicted results.

For different approaches, the order of the correctly predicted results is given in the table (the number in each element). For LP-BERT, the orders of the correctly predicted results are [1,2,1,2,2], while the orders are [3,3,5,3,3] for the MLM-based pre-training, and the orders are [6,21,12,6,4] without pre-training. The results suggest that LP-BERT provides superior performance compared to MLM-based pre-training and the model without pre-training. It is noteworthy that all presented results are typical and not cherry-picked for presentation, with the goal of avoiding misrepresenting the actual performance of the proposed method.

4.6 Significance Analysis

Significance analysis is performed in Figure 4, where we show the performance of each method at Hits@1, Hits@3, Hits@10, and Hits@100 metrics. It can be seen that in the region where N is relatively small (e.g., Hits@1), our method performs similarly compared to the representative work RotatE and HAKE based on graph embedding, but significantly better than the representative work StAR/KG-BERT/GenKGC based on language model embedding. In the region where N is relatively larger (e.g., Hits@10, Hits@100), our method is significantly better overall than...
the representative work RotatE/HAKE based on graph embedding and the representative work StAR/KG-BERT/GenKGC based on language model embedding. Compared to the StAR model, LP-BERT has a t-statistic and P-value of 1.26 and 0.21, respectively. Compared to the HAKE model, LP-BERT has a t-statistic and P-value of 8.21 and 6.3e-16, respectively. We note that the LP-BERT method has a very small P-value compared to the HAKE method. Taking the WN18RR dataset as an example, translation distance models represent entity meanings based on entity embedding. Except for entity relationships with strong predictive power, the prediction results of other unseen entities are almost random; i.e., the ordering of the ground-truth entities among all the entities in the candidate pool is more random compared to the semantic match models, so the distribution of all samples’ MR is poor overall. However, for the semantic match models, as they encode each wordpiece in the entity, the overall prediction ability is significantly better with much smaller MR metrics, although the recall of hits@1 and hits@3 is not as precise as that of translation distance models.

5 CONCLUSION AND FUTURE WORK

This article has addressed a crucial task within the natural language processing field, namely semantic network completion. The primary objective was to accurately predict the connections between entities in a KG’s semantic network. Our approach involved the utilization of a language model, LP-BERT, which incorporates multi-task pre-training and KG fine-tuning phases. During the pre-training phase, we introduced two innovative tasks, MEM and MRM, which aimed to facilitate the learning of both contextual knowledge and structural information inherent in the KG. This approach allowed LP-BERT to develop a comprehensive understanding of the underlying data. In the fine-tuning phase, we implemented a triple-style negative sampling technique within a batch, resulting in a substantial increase in the proportion of negative sampling, while minimizing any impact on training time. This approach significantly enhanced the efficiency and effectiveness of LP-BERT, as demonstrated through extensive experimentation on three diverse datasets. Moving forward, our future research will involve exploring additional diverse pre-training tasks and augmenting the model’s parameter size. These endeavors will enable LP-BERT to accommodate larger
graph knowledge, thereby expanding its potential applications and further advancing semantic network completion in natural language processing.

REFERENCES

[1] Bo An, Bo Chen, Xianpei Han, and Le Sun. 2018. Accurate text-enhanced knowledge graph representation learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 745–755.

[2] Yushi Bai, Zhitao Ying, Hongyu Ren, and Jure Leskovec. 2021. Modeling heterogeneous hierarchies with relationship-specific hyperbolic cones. In *International Conference on Neural Information Processing Systems*.

[3] Ivana Balážević, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor factorization for knowledge graph completion. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*. 5185–5194.

[4] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *International Conference on Management of Data*. 1247–1250.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *International Conference on Neural Information Processing Systems*.

[6] Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. *arXiv preprint arXiv:1906.05317* (2019).

[7] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. 2020. Low-dimensional hyperbolic knowledge graph embeddings. In *Annual Meeting of the Association for Computational Linguistics*.

[8] Yinhan Chen, Pasquale Minervini, Sebastian Riedel, and Pontus Stenetorp. 2021. Relation prediction as an auxiliary training objective for improving multi-relational graph representations. In *Conference on Automated Knowledge Base Construction (AKBC’21)*. DOI: 10.24432/C54K5W

[9] Louis Clonatre, Philippe Trempe, Amal Zouaq, and Sarath Chandar. 2020. MLMLM: Link prediction with mean likelihood masked language model. *arXiv preprint arXiv:2009.07058* (2020).

[10] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, Shijin Wang, and Guoping Hu. 2020. Revisiting pre-trained models for Chinese natural language processing. In *Conference on Empirical Methods in Natural Language Processing*.

[11] Yiming Cui, Wanxiang Che, Ting Liu, Bing Qin, and Ziqing Yang. 2021. Pre-training with whole word masking for chinese BERT. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), 3504–3514.

[12] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[14] Haipeng Gao, Kun Yang, Yuxue Yang, Rufai Yusuf Zakari, Jim Wilson Owusu, and Ke Qin. 2021. QuatDE: Dynamic quaternion embedding for knowledge graph completion. *arXiv preprint arXiv:2105.09002* (2021).

[15] Palash Goyal and Emilio Ferrara. 2018. Graph embedding techniques, applications, and performance: A survey. *Knowledge-based Systems* 151 (2018), 78–94.

[16] Charles Tapley Hoyt, Max Berrendorf, Mikhail Galkin, Volker Tresp, and Benjamin M. Gyori. 2022. A unified framework for rank-based evaluation metrics for link prediction in knowledge graphs. *arXiv preprint arXiv:2203.07544* (2022).

[17] Nicolas Hubert, Pierre Monnin, Armelle Brun, and Davy Monticolo. 2023. Enhancing knowledge graph embedding models with semantic-driven loss functions. *arXiv preprint arXiv:2303.00286* (2023).

[18] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics* 8 (2020), 64–77.

[19] Seyed Mehran Kazemi and David Poole. 2018. Simple embedding link for prediction in knowledge graphs. *Advances in Neural Information Processing Systems* 31 (2018).

[20] Jaejun Lee, Chanyoung Chung, and Joyce Jiyoung Whang. 2023. InGram: Inductive knowledge graph embedding via relation graphs. *arXiv preprint arXiv:2305.19887* (2023).

[21] Yeon-Chang Lee, JaeHyun Lee, Dongwon Lee, and Sang-Wook Kim. 2022. THOR: Self-supervised temporal knowledge graph embedding via three-tower graph convolutional networks. In *2022 IEEE International Conference on Data Mining (ICDM’22)*. IEEE, 1035–1040.

[22] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Roberta: A robustly optimized bert pretraining approach. In *International Conference on Learning Representations*. ACM Trans. Asian Low-Resour. Lang. Inf. Process., Vol. 22, No. 11, Article 250. Publication date: November 2023.
[23] Haonan Lu and Hailin Hu. 2020. DensE: An enhanced non-abelian group representation for knowledge graph embedding. arXiv preprint arXiv:2008.04548 (2020).

[24] Xin Lv, Yuxian Gu, Xu Han, Lei Hou, Juanzi Li, and Zhiyuan Liu. 2019. Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations. In Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing. 3376–3381.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. Advances in Neural Information Processing Systems 26 (2013).

[26] George A. Miller. 1998. WordNet: An Electronic Lexical Database. MIT Press.

[27] Jonas Mueller and Aditya Thiyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 30.

[28] Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. arXiv preprint arXiv:1906.01195 (2019).

[29] Dai Quoc Nguyen, Dat Quoc Nguyen, Tu Dinh Nguyen, and Dinh Phung. 2019. A convolutional neural network-based model for knowledge base completion and its application to search personalization. Semantic Web 10, 5 (2019), 947–960.

[30] Yanhui Peng and Jing Zhang. 2020. LineaRE: Simple but powerful knowledge graph embedding for link prediction. In ICDM. IEEE, 422–431.

[31] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP’14). 1532–1543.

[32] Jipeng Qiang, Yun Li, Yi Zhu, Yunhao Yuan, Yang Shi, and Xindong Wu. 2021. LSBert: Lexical simplification based on BERT. IEEE/ACM Transactions on Audio, Speech, and Language Processing 29 (2021), 3064–3076.

[33] Andrea Rossi, Denilson Barbosa, Donatella Firmani, Antonio Matinata, and Paolo Meriaaldo. 2021. Knowledge graph embedding for link prediction: A comparative analysis. ACM Transactions on Knowledge Discovery from Data 15, 2 (2021), 1–49.

[34] Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymansowska, Izabela Stefaniak, Michal Jarkiewicz, and Łukasz Okruszek. 2021. Detecting formal thought disorder by deep contextualized word representations. Psychiatry Research 304 (2021), 114135.

[35] Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In ESWC. Springer, 593–607.

[36] Jianhao Shen, Chenguang Wang, Ye Yuan, Jiawei Han, Heng Ji, Koushik Sen, Ming Zhang, and Dawn Song. 2022. PALT: Parameter-lite transfer of language models for knowledge graph completion. In Findings of the Association for Computational Linguistics: EMNLP 2022. Association for Computational Linguistics.

[37] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Ng. 2013. Reasoning with neural tensor networks for knowledge base completion. Advances in Neural Information Processing Systems 26 (2013).

[38] Tengwei Song, Jie Luo, and Lei Huang. 2021. Rot-pro: Modeling transitivity by projection in knowledge graph embedding. International Conference on Neural Information Processing Systems 34 (2021).

[39] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2018. RotatE: Knowledge graph embedding by rotational rotation in complex space. In International Conference on Learning Representations.

[40] Martin Sundermeyer, Hermann Ney, and Ralf Schlüter. 2015. From feedforward to recurrent LSTM neural networks for language modeling. IEEE/ACM Transactions on Audio, Speech, and Language Processing 23, 3 (2015), 517–529.

[41] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing text for joint embedding of text and knowledge bases. In Conference on Empirical Methods in Natural Language Processing. 1499–1509.

[42] Hung Nghiep Tran and Atsuhiko Takasu. 2022. MEIM: Multi-partition embedding interaction beyond block term format for efficient and expressive link prediction. arXiv preprint arXiv:2209.15597 (2022).

[43] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In International Conference on Machine Learning, PMLR, 2071–2080.

[44] Shikhar Vasishth, Soumya Sanyal, Vikram Nitin, and Partha Talukdar. 2019. Composition-based multi-relational graph convolutional networks. In International Conference on Learning Representations.

[45] Thanh Vu, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2019. A capsule network-based embedding model for knowledge graph completion and search personalization. In Conference on the North American Chapter of the Association for Computational Linguistics. 2188–2189.

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

[47] Haoyu Wang, Vivek Kulkarni, and William Yang Wang. 2018. Dolores: Deep contextualized knowledge graph embeddings. arXiv preprint arXiv:1811.00147 (2018).
[48] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. IEEE Transactions on Knowledge and Data Engineering 29 (2017), 2723–2743.

[49] Rui Wang, Bicheng Li, Shengwei Hu, Wenqian Du, and Min Zhang. 2019. Knowledge graph embedding via graph attenuated attention networks. IEEE Access 8 (2019), 5212–5224.

[50] 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.

[51] Han Xiao, Minlie Huang, Lian Meng, and Xiaoyan Zhu. 2017. SSP: Semantic space projection for knowledge graph embedding with text descriptions. In 31st AAAI Conference on Artificial Intelligence.

[52] Xin Xie, Ningyu Zhang, Zholeh Li, Shumin Deng, Hui Chen, Feiyu Xiong, Mosha Chen, and Huajun Chen. 2022. From discrimination to generation: Knowledge graph completion with generative transformer. In Companion of The Web Conference 2022, Frédérique Laforest, Raphael Troncy, Elena Simperl, Deepak Agarwal, Aristides Gionis, Ivan Herman, and Lionel Méndini (Eds.). ACM, 162–165.

[53] Jiacheng Xu, Kan Chen, Xipeng Qiu, and Xuanjing Huang. 2016. Knowledge graph representation with jointly structural and textual encoding. arXiv preprint arXiv:1611.08661 (2016).

[54] Wentao Xu, Zhiping Luo, Weiqing Liu, Jiang Bian, Jian Yin, and Tie-Yan Liu. 2021. KGE-CL: Contrastive learning of knowledge graph embeddings. arXiv preprint arXiv:2112.04871 (2021).

[55] 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:1412.6573 (2014).

[56] Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. arXiv:1909.03193 (2019).

[57] Shuanglong Yao, Dechang Pi, and Junfu Chen. 2022. Knowledge embedding via hyperbolic skipped graph convolutional networks. Neurocomputing 480 (2022), 119–130.

[58] Jason Youn and Ilias Tagkopoulos. 2023. KGLM: Integrating Knowledge Graph Structure in Language Models for Link Prediction. arxiv:2211.02744 [cs.CL].

[59] Mohamad Zamin, Hassan Reza, and Minou Rabiei. 2022. A review of knowledge graph completion. Information 13, 8 (2022), 396.

[60] Jiayui Zhang, Jian Huang, Jialong Gao, Runhai Han, and Cong Zhou. 2022. Knowledge graph embedding by logical-default attention graph convolution neural network for link prediction. Information Sciences 593 (2022), 201–215.

[61] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. Advances in Neural Information Processing Systems 32 (2019).

[62] Yichi Zhang, Mingyang Chen, and Wen Zhang. 2023. Modality-aware negative sampling for multi-modal knowledge graph embedding. arXiv preprint arXiv:2304.11618 (2023).

[63] Yongqi Zhang, Quanming Yao, Wenyuan Dai, and Lei Chen. 2020. AutoSF: Searching scoring functions for knowledge graph embedding. In International Conference on Data Engineering. IEEE.

[64] Zhanqiu Zhang, Jianyu Cai, and Jie Wang. 2020. Duality-induced regularizer for tensor factorization based knowledge graph completion. International Conference on Neural Information Processing Systems 33 (2020).

[65] Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In Proceedings of the AAAI Conference on Artificial Intelligence.

[66] 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, 798–807.

Received 2 March 2023; revised 21 August 2023; accepted 8 October 2023