Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 5968047, 12 pages
https://doi.org/10.1155/2022/5968047

Research Article

A Renovated CNN-Based Model Enhances KGC Task Performance

Fang Miao,[1] **Xueting Wang**,[2] **Feng Feng**,[3] **Cong Jin**,[1] **and Libiao Jin**[1]

[1]School of Information and Communication Engineering, Communication University of China, Beijing 10024, China
[2]China United Network Communications Corporation Qingdao Branch, Qingdao, China
[3]Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO 65211, USA

Correspondence should be addressed to Fang Miao; miaof03@cuc.edu.cn

Received 15 October 2021; Accepted 24 February 2022; Published 16 March 2022

Academic Editor: Bo Rong

Knowledge graph (KG) contains a large number of real-world knowledge and has become an invaluable aid to assist the application of artificial intelligence. Knowledge graph completion (KGC) is the task to complete the missing triple in KG database. Our goal in this study is to enhance the performance of KGC tasks based on CNN model. To do this, we first investigated the effect of adding multiple filters of different shapes into the pioneer model. The obscure improvement leads us to seek other approaches. Our second proposed model, termed DP-ConvKB, which is a deep convolution-neural-network-based model, outperforms state-of-the-art models on several metrics. Our study provides supporting evidence that, by cooperating deep pyramid network structure into models, it can significantly improve the KGC performances.

1. Introduction

Since the development of the Semantic Web in the 1980s, knowledge graph (KG), as a concept of knowledge base, serves as a surrogate of the real-world information, which focuses on describing the relationships between interlinked entities. Based on the graph-structured data model of integrating data, KG uses a triple (head entity, relation, and tail entity) to store interlinked descriptions of entities. Thanks to its structure-based and machine-readable attributes, the application of KG proliferates in various domains, such as search, analytics, and recommendation. In 2012, search engine giant Google launched their KG [1], and by May 2020, Google KG had grown to 500 billion facts on 5 billion entities [2]. However, information incompleteness is a problem that always exists in all KGs, including massive KGs. For example, in Freebase and DBpedia, more than 66% of the person entities are missing a birthplace [3, 4]. These missing facts not only affect the KG structure itself but also would sometimes result in inferring misleading information. Therefore, knowledge graph completion (KGC) researchers aim to identify missing links under the supervision of the existing KG. The specific work scope of KGC includes finding and mining missing entities and relationships [5], link prediction [6], and inferring new facts [7].

The key problem of knowledge graph completion is how to represent and model the combination of entities and relations. Data representation and representation learning are known to play a pivotal role in the development of KGC [8]. Embedding method is a popular data representation technique, which generally converts the entities and relations into low-dimensional vectors or matrixes. For example, in TransE [9], the candidate triple (h, r, t) is used for representing a valid fact, and then, the relation r corresponds to a translation between the embeddings of the head entity h and the tail entity t, that is, $h + r \approx t$. Several succeeding transition-based models are proposed with the basic idea of TransE, such as TransH [10], TransD [11], TransR [12], and TransG [13]. Among them, TransH models a relation as a hyperplane and TransR divides the workspace into entity space and relation space during representing the relations and entities. In addition, a trilinear dot product has also been applied to compute the score for each triple; DistMult [14] and ComplEx [15] are two typical such linear models.

KGC models can be generally divided into embedding-based models, linear models, and neural network models. Neural networks (NNs) have been widely used in machine learning tasks such as pattern recognition and perception science [16] and also gained more attention in the field of KG recently. By cooperating with convolution algorithm, convolutional neural networks (CNNs) are a specialized type of NNs that have been heavily used in computer vision since its birth. Such popularity of CNN-based applications in computer vision is because the neural-network-based structure of CNNs can match the composition of images natively. Since Kim proposed TextCNN in 2013 [17], CNNs have also begun to receive numerous attentions in the field of natural language processing (NLP), such as sentence classification [18], sentence modeling [17], and search query retrieval [19]. Most of these models adopt the convolution layer similar to TextCNN to extract features in the embedding. Inspired by computer vision, Dettmers et al. proposed the first CNN-based model for the KGC task—ConvE [20]. Following ConvE, Nguyen et al. proposed the model ConvKB [21], where the triple (h, r, t) is represented as k-dimensional vectors V_h, V_r, V_t, and the input matrix combines these vectors into a $k \times 3$ matrix. In the convolution layer, different filters with the same shape of 1×3 are designed to explore the global features among the same dimensional units of the embedding triple (v_h, v_r, v_t).

Although the model ConvKB has overcome the limitation of ConvE and obtains better link prediction results than the existing models on several benchmark datasets, there are still several unsolved problems. For example, limited by the convolution kernel of fixed size, long-distance interaction between different positions in embedding vectors cannot be extracted, but only shallow information from the single convolutional layer. Therefore, multiple methods have been proposed to solve this problem, such as CapsE [22] and ConvR [23]. In this paper, we applied a deep pyramid convolutional network structure with the original ConvKB model and designed a new model named deep pyramid ConvKB (DP-ConvKB).

We summarize the main contribution of our work as follows.

To improve KGC task performance, we first attempted it with a tentative model by introducing multiple complex filters to the ConvKB, hereafter referred to as multifilter ConvKB (MF-ConvKB). Compared to other existing CNN-based models, including ConvKB, experiment results on two benchmark datasets, FB15k-237 and WN18RR, showed that MF-ConvKB could only bring about mild improvements on some specific metrics. However, it motivated us to seek for other possible model structures which might potentially help to overcome the limit of improvement. We found that by incorporating a deep pyramid network structure into ConvKB, the new-designed model DP-ConvKB could significantly improve the KGC performances on several metrics.

2. Related Work

2.1. Embedding-Based Model. Embedding-based TransE was the first model using transseries methods for modeling multirelational data [9]. Inspired by the Word2Vec Skip-gram model, the TransE maps the relationship in the triple to the translations in latent feature space, denoted as $h + r \approx t$. However, the TransE model can only work well on the instances with one-to-one relation due to its lower parameter complexity and fails to deal with complex relations, such as one-to-more, more-to-one, and more-to-more. To enhance the scope ability and the efficiency of the score function of the previous models, TransH [10] is proposed to model the complex relations as a hyperplane together with a translation operation on it; however, this model still lacks of adaptation to scenarios with multirelations. TransD [11] and TransR/CTransR [12] represent entities and relations in separate spaces and project each entity with a relation-specific matrix. Unlike TransR, STransE [24] and TranSparse [25] tie the head and tail entities together with their own projection matrices. In addition, PTransE [26] and PTransR [27] take the relation path information into consideration when representing the triple (h, r, t). Recently, aiming at expanding the embedding space, embedding-based models like RotatE [28] continue to play a big role in the KGC task field.

2.2. Linear Model. RESCAL [29] is a typical bilinear model obtaining the underlying semantics information of the representation of the entity, but this model is prone to overfitting because of the large amounts of parameters. DistMult [14] can be considered as a special case of RESCAL, in which DistMult represents each relation as a diagonal matrix rather than a full matrix to avoid overfitting. ComplEx [15] and SimplE [30] can be viewed as direct extensions of the DistMult, in which ComplEx applies the complex domain to handle a variety of binary relations, and in SimplE, the subject and object embeddings for the same entity are learned dependently.

2.3. Neural Network Model. ConvE [20] and ConvKB [21] both are CNN-based models, and ConvE is the first model with CNNs applied for the KGC task. In ConvE, various filters with the fixed shape are operated over the input matrix—reshaping and concatenating of the head entity and relation embeddings. The design of the input is straightforward and one sided, which ignores the global information among the same dimensional position of the representation of the whole triple. ConvKB was proposed to deal with the problem from ConvE, replacing the reshaping operation with a convolution layer over the embedding triple (h, r, t). CapsE [22] applies the capsule network for the KGC task by adding a capsule network layer instead of the convolution layer at the base of the ConvKB. SACN [31] and R-GCN [32] are graph convolutional network- (GCN-) based models, and DOLORES [33] introduces long short-term memory (LSTM) to KGC tasks. In addition, KBGAT [34], GAATs [35], ConvR [23], and CoPER-ConvE [36] are effective methods which are proposed in recent years. Collectively, incorporating NNs has surfaced to become the mainstream way to solve the problem of the incomplete knowledge graph. Table 1 illustrates the score function and the optimization methods of each related work.

Table 1: The score function and the optimization methods of each related work. In these models, the entities and relations are represented by vectors h, r, t. And in the CNN-based model, g denotes a nonlinear function, $*$ denotes a convolution operator, $\cdot$ denotes a dot product, and Ω denotes a set of filters.

Category	Model	Score function	Opt.
Embedding-based model	TransE	$\|h + r - t\|_{1/2}$	SGD
	TransH	$\|h_\perp + r - t_\perp\|_{1/2}$; $h_\perp = h - w_r^\top \mathrm{t} w_r$; $t_\perp = t - w_r^\top \mathrm{t} w_r$	SGD
	TransD	$\|h_\perp + r - t_\perp\|_{1/2}$; $h_\perp = M_r h$, $t_\perp = M_r t$; $M_r \in \mathbb{R}^{k\times d}$	AdaDelta
	TransR	$\|h_\perp + r - t_\perp\|_{1/2}$; $h_\perp = M_r^1 h$; $t_\perp = M_r^2 t$; $M_r^1 = w_r w_h^\top + I$; $M_r^2 = w_r w_t^\top + I$	SGD
	STransE	$\left\|W_{r,1} h + r - W_{r,2} t\right\|_{1/2}$; $W_{r,1}, W_{r,1} \in \mathbb{R}^{k\times k}$	SGD
	TranSparse	$\|h_\perp + r - t_\perp\|_{1/2}$; $h_\perp = M_r^1(\theta_r^1) h$; $t_\perp = M_r^2(\theta_r^2) t$	SGD
Linear model	RESCAL	$h_r^\top M_r t$; $M_r \in \mathbb{R}^{d\times d}$	SGD
	DistMult	$h_r^\top M_r t$; $M_r = \mathrm{diag}\,(r)$; $\mathrm{r} \in \mathbb{R}^d$	AdaDelta
	ComplEx	$\mathrm{Re}\,(h^\top \ \mathrm{diag}\,(r)\bar{t})$ $\mathrm{Re}\,(.)$: the real part of a complex value $\bar{t}$: the conjugate of t	AdaDelta
	SimplE	$\frac{1}{2}\left(h \odot r \cdot t + t \odot r^{-1} \cdot h\right)$ $\odot$ represents Hadamard multiplication $\cdot$represents dot product	AdaDelta
Neural network model	ConvE	$g\left(\mathrm{vec}\left(g\left(\mathrm{concat}\left[\bar{h}\,;\bar{r}\right] * \Omega\right)\right) W\right) \bullet t$	Adam
	ConvKB	$\mathrm{concat}(g([h\,; r\,; t] * \Omega)) \bullet w$	Adam
	CapsE	$\|\mathrm{capsnet}(g([h\,; r\,; t] * \Omega))\|$	Adam

Deep CNNs are commonly used to extract depth features from images, and a number of powerful models have emerged, such as LeNet [37], AlexNet [38], ResNet [39], and Google Inception Net [40]. Inspired by ResNet, Johnson and Zhang proposed a low-complexity word-level deep convolutional neural network architecture for text categorization—DPCNN [41]. In this paper, we proposed a novel neural network model, namely, DP-ConvKB, which takes the advantage of deep convolution structure, to improve the performance of KGC.

3. The Construction of Proposed Models

As we mentioned above, we attempted two different approaches to improve the KGC task performances by ConvKB. In the first model, MF-ConvKB, we incorporated multiple filters of different shapes into ConvKB. In the second model, DP-ConvKB, we utilized a deep pyramid convolutional network to improve the KGC task performance. In this section, we present the procedures for constructing these two models.

3.1. MF-ConvKB. An incomplete KG ζ collects the facts in the form of (h, r, t), with $h, t \in \mathscr{E}, r \in \mathscr{R}$, where $\mathscr{E}$ and $\mathscr{R}$ denote the sets of entities and relations, respectively. Each triple is represented as a unique embedding group (v_h, v_r, v_t) collecting with K-dimensional vectors, and all these vectors are concatenated into a matrix $A \in \mathbb{R}^{k\times 3}$ and $A_i \in \mathbb{R}^{1\times 3}$ represents the i-th row of A. For example, a filter $\omega \in \mathbb{R}^{h\times 3}$ is applied to the window consisting from A_i to A_{i+h-1} to generate a feature c_i, which is defined as follows:

$$c_i = g(\omega \bullet A_{i:i+h-1} + b), \tag{1}$$

where $b \in \mathbb{R}$ is a bias term and g is a nonlinear function such as rectified linear unit (ReLU), and we use $c = [c_1, c_2, \cdots, c_{k-h+1}]$ to form the feature map of this filter.

Different from the ConvKB that only uses filters of a single shape, our model involves multiple shapes of filters to generate the transitional features from the pretrained embeddings. As shown in Figure 1, multiple filters are designed to capture features from long or short distances. We used $N = [N_1, N_2, N_3]$ to denote the number of filters of three different shapes, which are 1×3, 2×3, and 3×3, so the original model ConvKB can be viewed as a special case of MF-ConvKB with $N_2 = N_3 = 0$. After convolution, all $\sum_{i=1}^{3} N_i$ feature maps are concatenated into a vector V. Then, the score for the triple (h, r, t) is calculated by a dot product between V and w, where $w \in \mathbb{R}^{\sum_{i=1}^{3} N_i \times 1}$ is a weight vector, and the score function f is defined as in

$$V_i = \mathrm{concat}(g([v_h\,; v_r\,; v_t] * \Omega_i)), \tag{2}$$

$$f(h, r, t) = \mathrm{concat}(V_1, V_2, V_3) \cdot w, \tag{3}$$

where Ω_i denotes the set of filters of the i-th shape, V_i is the feature map generated from the set of filters of the i-th

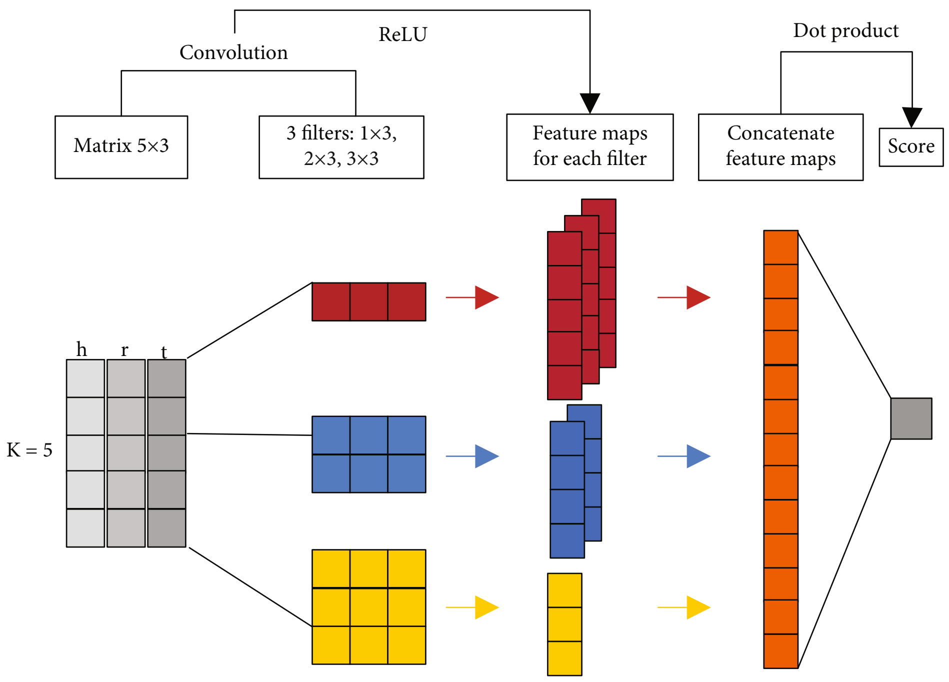


FIGURE 1: Illustration of the MF-ConvKB model framework.

shape, $*$ denotes a convolution operator, and $\cdot$ denotes a dot product.

3.2. DP-ConvKB. Compared to the tentative MF-ConvKB model, we implemented a completely different approach to extract features in DP-ConvKB. Instead of adding a variety of filters, we took advantage of the deep residual technique [39] and built another module of deep residual learning in addition to the pioneer ConvKB model. In this way, more features in a long distance can be taken into consideration. DP-ConvKB takes its name from referring its structure to deep pyramid (DP) CNNs [41]. As shown in Figure 2, the design of DP-ConvKB consists of two modules in general.

3.2.1. ConvKB Module. The left module is from the original ConvKB, where TransE model is used to initialize the input entities and relations. In order to keep the maximum potential of the transitional features in the transition-based model, we only applied the filter $w \in \mathbb{R}^{1\times 3}$ to exploit the global relationships among the same dimensional entries of the embedding triple. The feature map V_{ConvKB} is defined as

$$V_{\text{ConvKB}}(h, r, t) = \text{concat}(g([v_h ; v_r ; v_t]) * \Omega), \quad (4)$$

in which Ω denotes the set of filters and $*$ denotes a convolution operator. The output of the final feature map from this module is fed into the right module as the initial layer of region embedding.

3.2.2. Deep Residual Learning Module. In the deep residual learning module, the initial region embedding is followed by two convolutional layers, an activation layer, and a shortcut connection. This module with shortcut connection can be represented as $f(x) + x$, where f denotes the skipped layers of feedforward neural networks and x is the shortcut connection as proposed in ResNet [39]. A similar structure has also been applied in DPCNN for text categorization [41]. This design of the shortcut connection can effectively avoid the vanishing gradient problem when training the deep NNs. ReLU was chosen as the activation layer, which corresponds to the design of the feature map.

In this paper, we focused more on the global information among the same dimension between long distances, but different from the semantic features among a long sentence; in the KGC task, the input matrix is only composed of the embeddings of entities and relations, without the semantic characters between different dimensions. In order to capture the features with a relatively long distance, we used the kernel size of 2 for convolution, as shown in Figure 2.

After the first shortcut connect, another circulation block is added, and it consists of a pooling layer, two convolution layers, and a shortcut connection, in which the convolution operation is defined as $g(x) * \Omega + b$, where $g(x)$ is for ReLU and Ω is the filter initialized by a normal distribution. At the beginning of this submodule, we performed max-pooling with size 2 and stride 2: the pooling layer chooses the maximum over 2 contiguous internal vectors and the 2-stride pooling window reduces the size of the input feature map by half. After this, output from each pooling layer is collected, arranged in the form of a "pyramid," and this is where DP takes its name.

For the shortcut connection $f(x) + x$, both $f(x)$ and x require the same dimensionality so that they can be summed up. To avoid the extra dimension matching operation, in the DP structure, we unified the convolutional layers with the same number of filters to obtain N_* feature maps for each convolution layer. In DP-ConvKB, we set $N_* = N$, where N is the number of filters used in ConvKB (the right module);

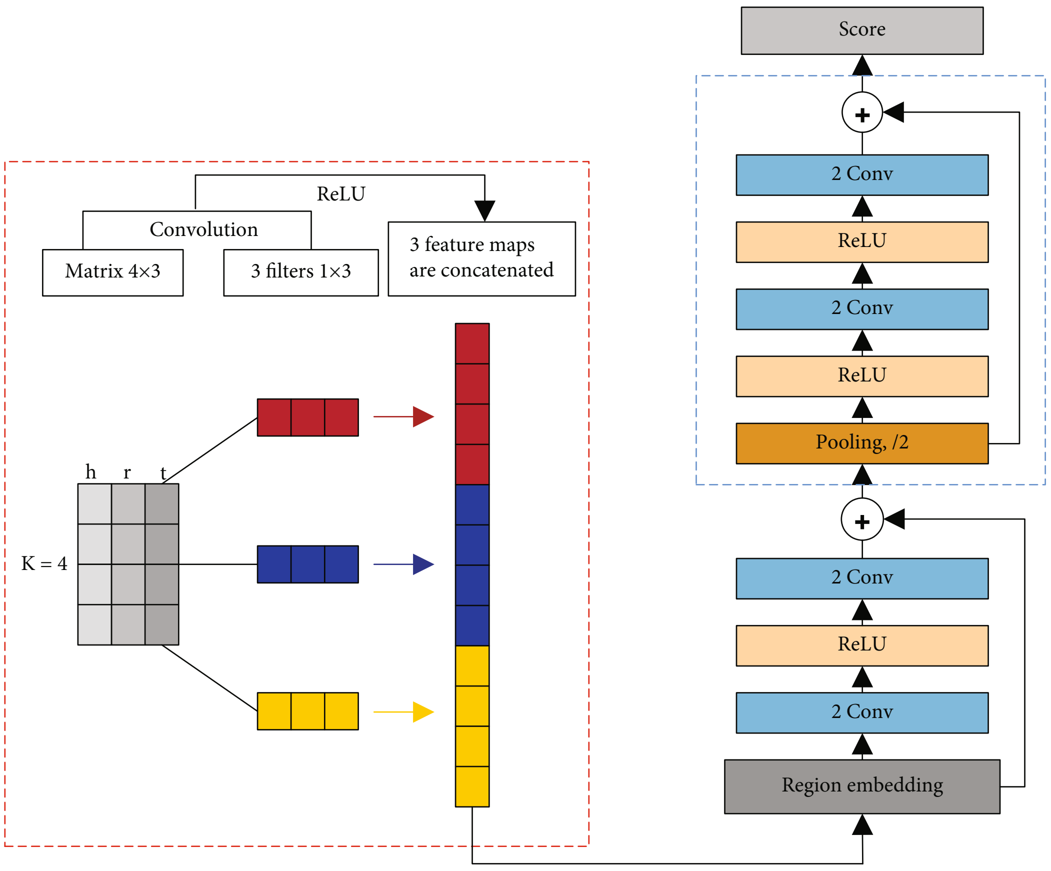


Figure 2: Illustration of the DP-ConvKB model framework.

Table 2: Statistics of the experimental datasets used in this study. #Entity is the number of entities, #Rel is the number of relation types, and #Train, #Valid, and #Test are the numbers of triples in the training, validation, and test sets, respectively.

Dataset	#Entity	#Rel	#Train	#Valid	#Test
WN18RR	40943	11	86835	3034	3134
FB15k-237	14541	237	272115	17535	20466

the value of region embedding is the same as V_{ConvKB}. After the circulation block, the score function of the model can be written as follows:

$$f_{\text{DP-ConvKB}} = V_C \cdot W, \quad (5)$$

where V_C is the output of the final convolution layer and W is a weight vector.

4. Experiment Setup

4.1. Dataset. We evaluated MF-ConvKB and DP-ConvKB models on two benchmark datasets: WN18RR and FB15k-237, which are the subsets of the datasets WN18 and FB15k, respectively. This is because WN18 and FB15k contain highly redundant relations, as mentioned in studies [42, 43]; these highly redundant relations could lead to inaccurate test results. One example of redundant relations is when a triple and its inverse triple exist in the test set and training set, respectively. This would make the original triple easily retrievable which leads to an overly good result and jeopardize the model's generalization. WN18RR and FB15k-237 are two datasets intentionally designed to remove triples with such inverse relation. Statistics of the two benchmark datasets are given in Table 2.

The relations from the two datasets can be generally classified into 4 categories: $1-\text{to}-1$, $1-\text{to}-M$, $M-\text{to}-1$, and $M-\text{to}-M$, with M for MANY. According to [9], $1-\text{to}-1$ denotes one head can appear with one tail entity at most, and $1-\text{to}-M$ denotes one head can appear with more than one tail. Similarly, $M-\text{to}-1$ denotes more than one head can appear with the same tail, and $M-\text{to}-M$. is for the case that multiple heads map to multiple tails. We found that 0.9% of the relations on FB15k-237 is $1-\text{to}-1$ type, and for $1-\text{to}-M$, $M-\text{to}-1$, and $M-\text{to}-M$ type, the portion is 6.3%, 20.5%, and 72.3%, respectively. Regarding the WIN18RR dataset, there are 11 relations, among which, *also_see*, *similar_to*, *verb_group*, and *derivationally_*

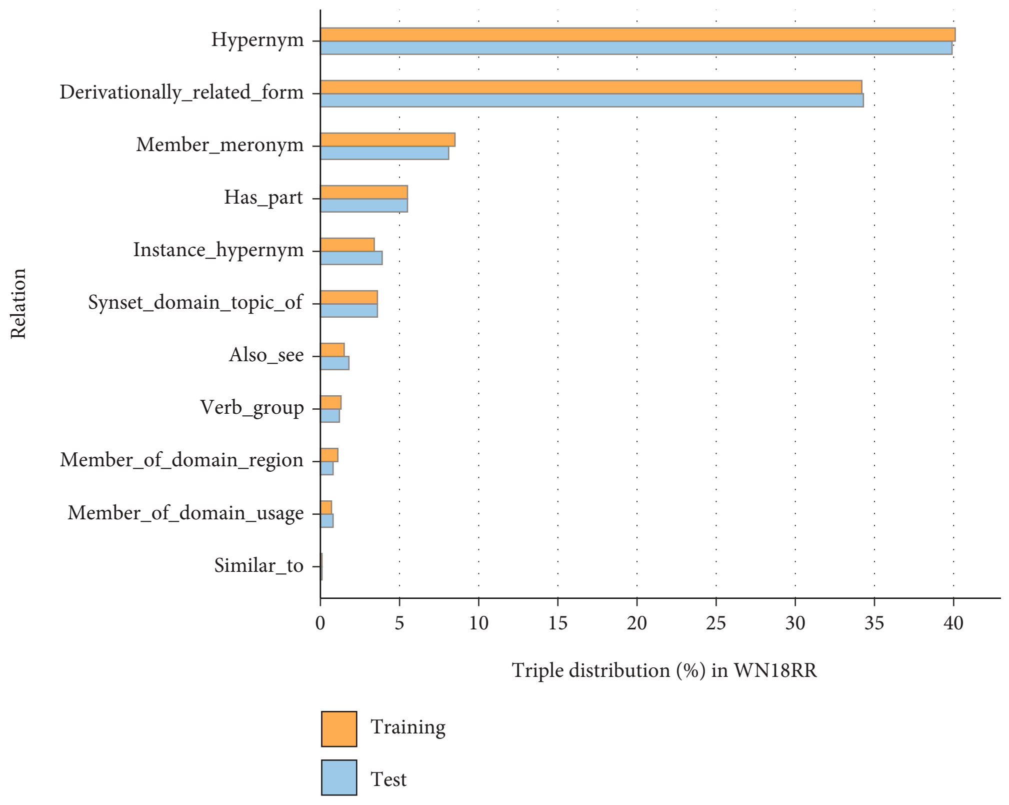


Figure 3: Distribution of both training and test set in WN18RR based on relations.

related_form can be classified into $M-\text{to}-M$ relation, and *member_meronym* and *hypernym* are $1-\text{to}-M$ and $M-\text{to}-1$ relation, respectively. Moreover, there are 1251 triples containing the relation *hypernym* and 1074 triples containing the relation *derivationally_related_form*, which in total accounts for about 75% of 3134 test triples in the test set. Figure 3 shows the percentage of each relation in both test and training sets.

4.2. Design of Loss Function. The loss function was minimized as follows:

$$L = \sum_{(h,r,t)\in\varsigma\cup\varsigma'} \log\left(1+\exp\left(l_{(h,r,t)}\cdot f_{\text{DP-ConvKB}}(h,r,t)\right)\right) + \frac{\lambda}{2}\|w\|_2^2,$$
$$\text{in which, } l_{(h,r,t)} = \begin{cases} 1 & \text{for}(h,r,t)\in\varsigma, \\ -1 & \text{for}(h,r,t)\in\varsigma', \end{cases} \tag{6}$$

where ς is the train set collection of valid triples, while ς' is a collection of invalid triples generated by corrupting a valid triple $(h,r,t)\in\varsigma$ by replacing the head entity or the tail entity with other entities in $\mathscr{E}$. According to Bernoulli trick [4], the new invalid triples (h',r,t) and (h,r,t') occur with the probability $\eta_h/(\eta_h+\eta_t)$ and $\eta_t/(\eta_h+\eta_t)$, respectively. For a certain relation r, η_h denotes the average number of head entities per tail entity, and η_t denotes the average number of tail entities per head entity. $f_{\text{DP-ConvKB}}$ is the score function as in (5), and $(\lambda/2)\|w\|_2^2$ is the $L2$ regularization on w.

4.3. Evaluation Protocol. The object of this study is to predict the missing entity in a triple, i.e., predicting the head entity with given (r,t) or predicting the tail entity with given (h,r). To evaluate our proposed models' performance, we focused on ranking the scores of candidate entities from the dataset. We employed three commonly used evaluation metrics in the previous study: mean rank (MR), mean reciprocal rank (MRR), and Hits@10. MR denotes the mean rank of all test triples as calculated in (7). It evaluates whether all of the ground-truth relevant items selected by the model are ranked higher or not.

$$\text{Mean Rank} = \frac{1}{N}\sum_{i=1}^{N}\text{rank}_i. \tag{7}$$

MRR is calculated by taking the mean of the reciprocal rank for each query of the test triples as in (8), and it only cares about the single highest-ranked relevant item.

$$\text{Mean Reciprocal Rank} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{\text{ran k}_i}. \tag{8}$$

Hits@k is the proportion of the rank that is lower than or equal to k, with k usually set to 10 in the link prediction task. Either lower MR, higher MRR, or higher Hits@k indicates a better result of the task. Note that, according to the work of

Table 3: Comparison results of link prediction task on WN18RR and FB15k-237 test sets. Hits@10 is reported in %.

Method	WN18RR			FB15k-237		
	MR	MRR	Hits@10	MR	MRR	Hits@10
TransE	3384	0.226	50.1	347	0.294	46.4
DistMult	5110	0.425	49.1	254	0.241	41.9
ComplEx	5261	0.444	50.7	339	0.247	42.8
ConvE	4187	0.433	51.5	244	0.325	50.1
ConvKB	1711	0.251	52.9	246	0.407	52.7
CapsE	719	0.415	56.0	303	0.523	59.3
RotatE	3340	0.476	57.1	177	0.338	53.3
MF-ConvKB	1273	0.261	56.1	207	0.307	48.2
DP-ConvKB	139	0.703	73.1	101	0.734	75.0

TransE [9], we had to remove the corrupt triples that already exist in the datasets when ranking the test triples.

4.4. Implementation Details. Prior to the model training, the entities and relation embeddings were first preprocessed by the embeddings produced from TransE. We then applied stochastic gradient descent (SGD) algorithm with 3000 epochs on WN18RR and FB15k-237 to train the TransE model parameters. We adopted grid search algorithm on the validation set, in order to optimize parameters, including the dimensionality of the word embedding, the margin hyperparameter, and the SGD learning rate. We fixed the batch size = 256 in our objective function. Our experiment showed that, for MF-ConvKB, the highest Hits@10 can be obtained when $k = 50$, $r = 5$, and $\eta = 5e^{-4}$ on WN18RR and $k = 100$, $r = 1$, and $\eta = 5e^{-4}$ on FB15k-237; for DP-ConvKB, these optimized parameters were $k = 100$, $r = 1$, and $\eta = 5e^{-4}$ for both datasets.

We used the Adam optimizer to train both MF-ConvKB and DP-ConvKB. For training MF-ConvKB, we set the initial learning rate of Adam [44] at $\eta \in \{1e^{-3}, 1e^{-4}, 5e^{-5}, 5e^{-6}\}$, and set the L_2-regularizer λ at 0.001 to avoid overfitting. After convolution, we chose ReLU as the activation function. We designed the filters with the shape of 1×3, 2×3, and 3×3. Each filter was initialized by a truncated normal distribution. The highest Hits@10 scores were obtained when using $N = [500{,}500{,}500]$, $k = 50$, and $\eta = 1e^{-4}$ on WN18RR and $N = [100{,}40{,}20]$, $k = 100$, and $\eta = 5e^{-6}$ on FB15k-237. For training DP-ConvKB, we set the initial learning rate of Adam at $\eta \in \{1e^{-5}, 1e^{-6}, 1e^{-7}, 5e^{-8}\}$ and set the L_2-regularizer λ at 0.001. To get the feature map for deep convolution, we initialized the original filter by a truncated normal distribution, and we fixed its shape at 1×3. In the deep NN part, convolutional filters were initialized by a normal distribution. The highest Hits@10 scores were obtained when using $N = 200$, $k = 100$, and $\eta = 5e^{-8}$ on WN18RR and $N = 100$, $k = 100$, and $\eta = 1e^{-7}$ on FB15k-237.

4.5. Main Results. Table 3 shows the link prediction comparison results between our two models and the state-of-the-art models.

From Table 3, it is clear that the proposed model DP-ConvKB significantly outperformed all other models listed, including the original ConvKB (baseline model, hereafter). Specifically, on dataset WN18RR, DP-ConvKB obtains significant improvements of $0.703 - 0.251 = 0.452$ in MRR and $73.1\% - 52.9\% = 20.2\%$ absolute improvement in Hits@10. On dataset FB15k-237, these improvements were 0.327 in MRR and 22.3% in Hits@10.

As shown in Table 3, we also noticed that the tentative model MF-ConvKB performed just slightly better than the baseline model on all metrics on WN18RR and even obtained a less score of MRR and Hits@10 on FB15k-237. However, it could achieve lower MR than the baseline model on both datasets. The shape of the convolutional filter is known to play a pivotal role in extracting features in fields like image processing [45], speech recognition [46], and sentence classification [47]. However, its effect in the link prediction task is not well understood. To investigate, we embedded convolutional filters with different shapes into MF-ConvKB and test model performances on link prediction tasks. Results are shown in Table 4: the first row with a single 1×3 filter represents the baseline model, and starting from the second row, we replaced the filter with different shapes and the final row was from MF-ConvKB. These results suggested that, in general, replacing the single filter in the baseline model with different shapes did not bring substantial improvements and combing multiple different filters can ameliorate performance on MR but failed on MRR and Hits@10 necessarily. Specifically, MF-ConvKB with a variety of filters can have better performances on WN18RR but not necessarily on FB15k-237. Taken together, these results indicate that combining filters of different shapes may not be the most efficient strategy for link prediction task performance improvement. Therefore, we sought other possible structures, e.g., deep NN in our proposed model DP-ConvKB, to improve link prediction task performance.

Figure 4 shows a result comparison between our proposed DP-ConvKB model and baseline model, tested on MRR and Hits@10 scores of different relations on dataset FB15k-237. Results of predicting head and tail entities are separately displayed. It is clear that DP-ConvKB improves baseline performances on all relations. Specifically, $M - 1$ achieves the highest scores, followed by $M - M$, $1 - 1$, and $1 - M$ ranking last. Moreover, DP-ConvKB shows more

Table 4: Link prediction results of different designed filters on WN18RR and FB15k-237 test sets.

Filter shape	WN18RR			FB15k-237		
	MR	MRR	Hits@10	MR	MRR	Hits@10
1×3 (baseline)	1711	0.251	52.9	246	0.407	52.7
2×3	1614	0.258	54.9	260	0.417	52.6
3×3	1547	0.251	54.9	277	0.424	52.6
$2 \times 3, 3 \times 3$	1480	0.256	55.0	296	0.400	50.8
$1 \times 3, 2 \times 3, 3 \times 3$ (MF-ConvKB)	1273	0.261	56.1	207	0.307	48.2

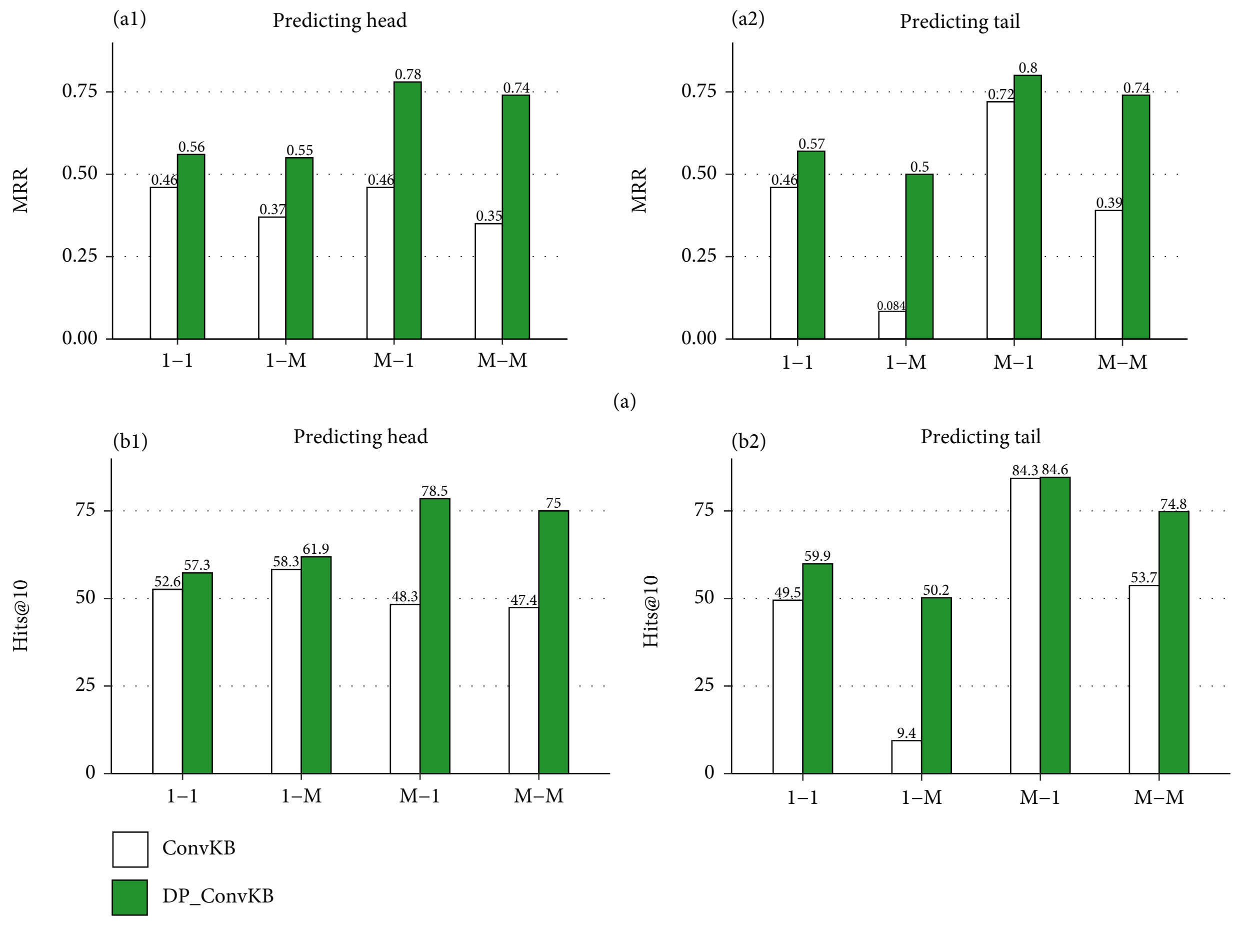
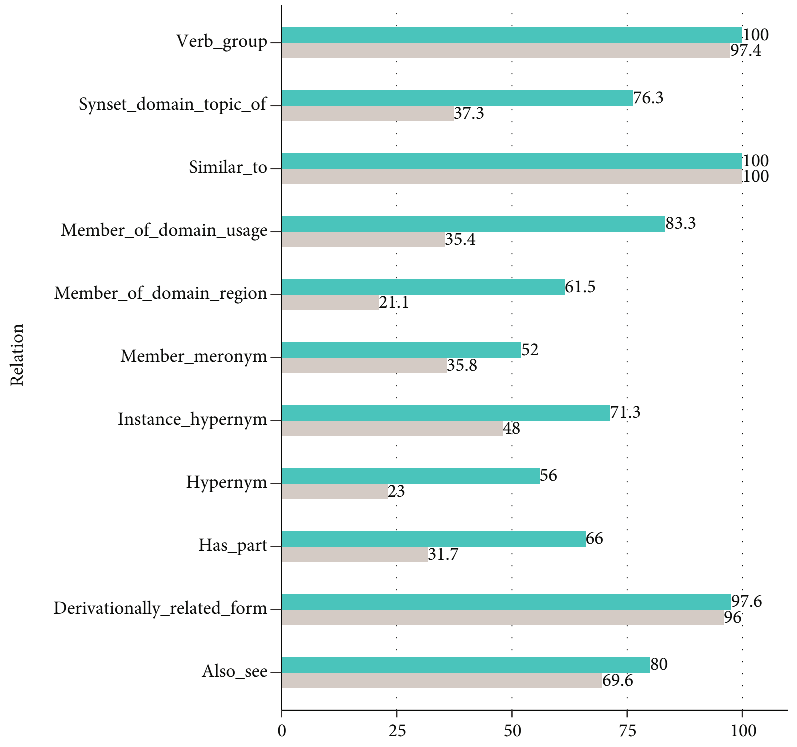


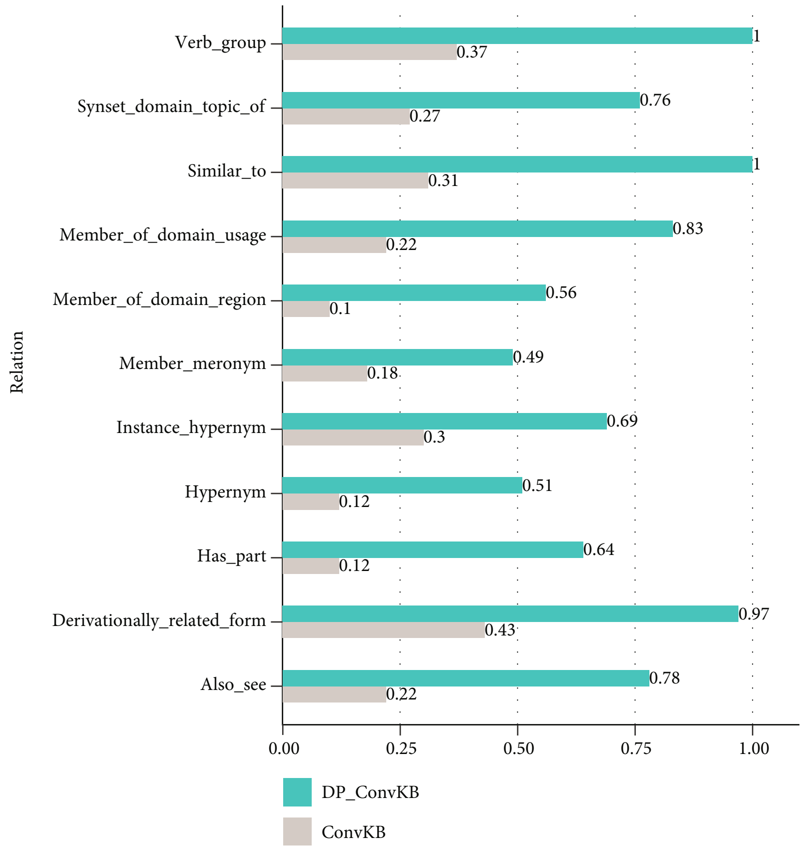
Figure 4: Performance of (a) MRR and (b) Hits@10 on FB15k-237 from DP-ConvKB and baseline model. A1 and B1 are the results of predicting head task, while A2 and B2 are the results of predicting tail task. Results are depicted in 4 categories of relations, which are $1 - 1$, $1 - M$, $M - 1$, and $M - M$.

robust performances when predicting head and tail on $1 - M$ and $M - 1$ relations, while the performances from the baseline model fluctuate in these cases (from 0.37 to 0.084 and 0.46 to 0.72 on MRR).

Performance comparisons for all relations on dataset WN18RR between DP-ConvKB and baseline model are shown in Figure 5. In general, our proposed model DP-ConvKB outperforms the baseline model in all considered relations on both Hits@10 and MRR metrics. Strikingly, when evaluated by MRR score, model performance is completely boosted by DP-ConvKB by more than 2 times. Particularly, performances on $M - M$ relations of *Verb_ group* and *similar_to* reach the highest scores on MRR. On WN18RR, we also tested the model's performances in predicting heads and tails. Figure 6 depicts the performance of DP-ConvKB in predicting head and tail tasks with the metrics of MRR and Hits@10 on the dataset WN18RR. We can see that the scores of predicting head and tail on almost every relation category are very close. As a comparison, the baseline model has a relatively large gap in predicting head and tail tasks which can be found in Figure 4. Results shown in Figure 6 demonstrate the

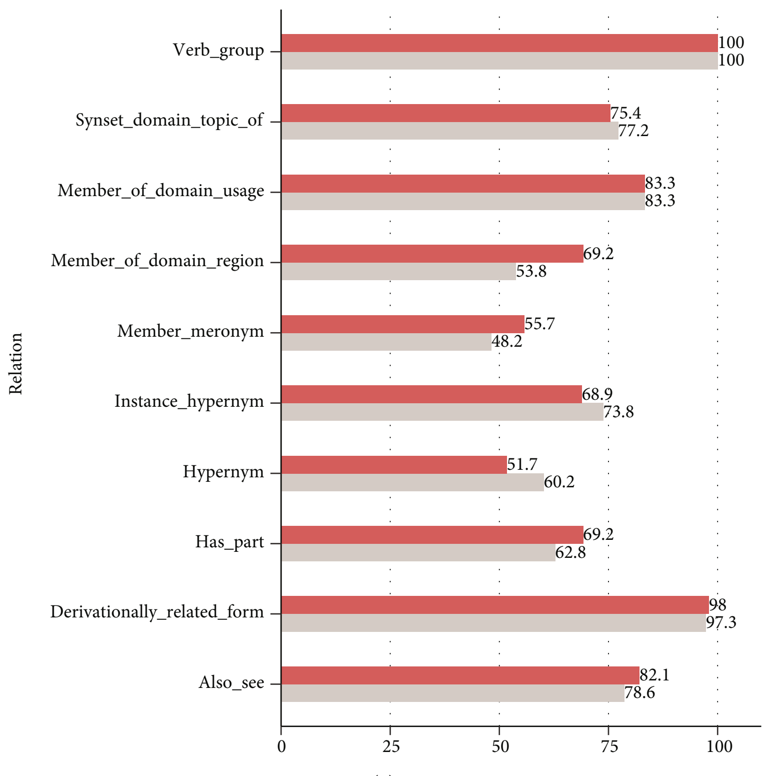


(a)

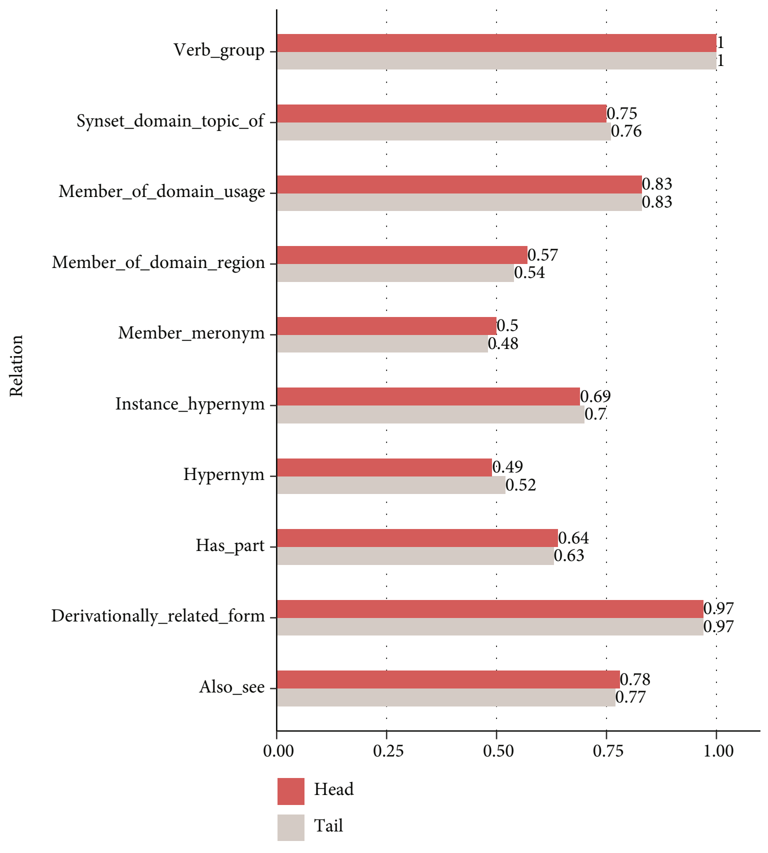


(b)

Figure 5: Performance of (a) Hits@10 and (b) MRR on WN18RR from DP-ConvKB and baseline model. There are 11 different relations in dataset WN18RR.



(a)



(b)

Figure 6: Performance of DP-ConvKB in predicting head and tail tasks on WN18RR. Evaluation metrics are (a) Hits@10 and (b) MRR.

ability of minimizing discrepancies in predicting head and tail when using DP-ConvKB.

Taken together, the overall experimental results show that the deep CNN-based model DP-ConvKB can effectively sum up the global features beyond distance and obtain the best performance for both the head prediction task and tail prediction task.

5. Conclusion

In this paper, we proposed a CNN-based model, DP-ConvKB, to improve the performance of the knowledge graph completion task. We first showed that simply adding a variety of filters into the pioneer model ConvKB might not be in the right direction to enhance its performance. We then designed a new model, DP-ConvKB, which cooperates a deep pyramid neural network into ConvKB; therefore, it is capable of exploring the deep features. Our results on datasets WN18RR and FB15k-237 show that DP-ConvKB outperforms the baseline model (ConvKB). DP-ConvKB obtains the best mean rank and the highest mean reciprocal rank and Hits@10. To sum up, our study demonstrates that, by implementing such deep convolutional network structure into models for KGC tasks, it can significantly improve the performances.

Although DP-ConvKB has achieved great improvement on the link prediction task, however, it involves more computational complexity and is difficult to find the optimized hyperparameters in the training process. In future work, we plan to prune the deep neural networks to minimize the training time and generalize our structure to other NLP tasks.

Data Availability

The data that support the findings of this study are available from the authors. Requests for access to these data should be made to Xueting Wang, xuetingcuc@163.com.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

References

[1] A. Singhal, "Introducing the knowledge graph: things, not strings," 2012, http://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html.

[2] D. Sullivan, "A reintroduction to our knowledge graph and knowledge panels," 2020, https://blog.google/products/search/about-knowledge-graph-and-knowledge-panels/.

[3] X. Dong, E. Gabrilovich, G. Heitz et al., *Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion*, ACM, 2014.

[4] D. Krompaß, S. Baier, and V. Tresp, "Type-constrained representation learning in knowledge graphs," in *The Semantic Web-ISWC*, vol. 9366, pp. 640–655, Springer, Cham, 2015.

[5] G. He, J. Li, W. X. Zhao, P. Liu, and J. R. Wen, "Mining implicit entity preference from user-item interaction data for knowledge graph completion via adversarial learning," in *Proceedings of The Web Conference*, Taipei Taiwan, 2020.

[6] A. Rossi, D. Firmani, A. Matinata, P. Merialdo, and D. Barbosa, "Knowledge graph embedding for link prediction," *A Comparative Analysis*, vol. 15, no. 2, pp. 1–49, 2021.

[7] Z. Chen, Y. Wang, B. Zhao, J. Cheng, and Z. Duan, "Knowledge graph completion: a review," *IEEE Access*, vol. 8, pp. 192435–192456, 2020.

[8] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: a review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[9] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," *Advances in Neural Information Processing Systems*, vol. 26, pp. 2787–2795, 2013.

[10] J. Zhang, "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the AAAI Conference on Artificial Intelligence*, Québec City, Québec, Canada, 2014.

[11] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proceedings of the 53rd annual meeting of the association for computational linguistics and the 7th international joint conference on natural language processing*, pp. 687–696, Beijing, China, 2015.

[12] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," *Twenty-ninth AAAI conference on artificial intelligence*, 2015.

[13] X. Han, M. Huang, H. Yu, and X. Zhu, "TransG: a generative mixture model for knowledge graph embedding," 2015, https://arxiv.org/abs/1509.05488.

[14] B. Yang, W. T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," 2014, https://arxiv.org/abs/1412.6575.

[15] T. Trouillon, J. Welbl, S. Riedel, R. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," 2016, https://JMLR.org.

[16] M. Yan, Z. Li, X. Yu, and C. Jin, "An end-to-end deep learning network for 3D object detection from RGB-D data based on Hough voting," *IEEE Access*, vol. 8, pp. 138810–138822, 2020.

[17] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, "A convolutional neural network for modelling sentences," vol. 1, 2014, https://arxiv.org/abs/1404.2188.

[18] Y. Kim, "Convolutional Neural Networks for Sentence Classification," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* of *Association for Computational Linguistics*, pp. 1746–1751, Doha, Qatar, 2014.

[19] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil, "Learning semantic representations using convolutional neural networks for web search," in *Proceedings of the 23rd International Conference on World Wide Web*, Seoul, Korea, 2014.

[20] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *32nd AAAI Conference on Artificial Intelligence (AAAI-18), February 2018*, New Orleans, LA, USA, 2017.

[21] Q. N. Dai, D. N. Tu, D. Q. Nguyen, and D. Phung, "A novel embedding model for knowledge base completion based on convolutional neural network," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, New Orleans, Louisiana, 2018.

[22] D. Q. Nguyen, T. Vu, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A capsule network-based embedding model for knowledge graph completion and search personalization," in *Proceedings of the 2019 Conference of the North*, Minneapolis, Minnesota, 2019.

[23] X. Jiang, Q. Wang, and B. Wang, "Adaptive convolution for multi-relational learning," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 978–987, Minneapolis, Minnesota, June 2019.

[24] D. Q. Nguyen, K. Sirts, L. Qu, and M. Johnson, "STransE: a novel embedding model of entities and relationships in knowledge bases," 2016, https://arxiv.org/abs/1606.08140.

[25] G. Ji, K. Liu, S. He, and J. Zhao, "Knowledge graph completion with adaptive sparse transfer matrix," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, Arizona, 2016.

[26] Y. Lin, Z. Liu, H. Luan, M. Sun, S. Rao, and S. Liu, "Modeling relation paths for representation learning of knowledge bases," 2015, https://arxiv.org/abs/1506.00379.

[27] W. Huang, L. Ge, and J. Zhi, "Improved knowledge base completion by the path-augmented TransR model," in *International Conference on Knowledge Science, Engineering and Management*, Springer, Cham, 2017.

[28] Z. Sun, Z. H. Deng, J. Y. Nie, and J. Tang, "RotatE: knowledge graph embedding by relational rotation in complex space," 2019, https://arxiv.org/abs/1902.10197.

[29] M. Nickel, V. Tresp, and H. P. Kriegel, "A three-way model for collective learning on multi-relational data," *International Conference on International Conference on Machine Learning*, , Omnipress, Madison, WI, USA, 2011.

[30] S. Mehran Kazemi and D. Poole, "SimplE embedding for link prediction in knowledge graphs," 2018, https://arxiv.org/abs/1802.04868.

[31] C. Shang, Y. Tang, J. Huang, J. Bi, and B. Zhou, "End-to-end structure-aware convolutional networks for knowledge base completion," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3060–3067, 2019.

[32] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. V. Berg, and M. Welling, "Modeling relational data with graph convolutional networks," in *European Semantic Web Conference*, Springer, Cham, 2018.

[33] H. Wang, V. Kulkarni, and W. Y. Wang, "DOLORES: deep contextualized knowledge graph embeddings," 2018, https://arxiv.org/abs/1811.00147.

[34] D. Nathani, J. Chauhan, C. Sharma, and M. Kaul, "Learning attention-based embeddings for relation prediction in knowledge graphs," 2019, https://arxiv.org/abs/1906.01195.

[35] R. Wang, B. Li, S. Hu, W. Du, and M. Zhang, "Knowledge graph embedding via graph attenuated attention networks," *IEEE access*, vol. 99, pp. 1–1, 2019.

[36] G. Stoica, O. Stretcu, E. A. Platanios, T. Mitchell, and B. Póczos, "Contextual parameter generation for knowledge graph link prediction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 3, pp. 3000–3008, 2020.

[37] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[39] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, Las Vegas, NV, USA, 2016.

[40] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Boston, MA, 2014.

[41] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 562–570, Vancouver, Canada, 2017.

[42] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon, "Representing text for joint embedding of text and knowledge bases," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, pp. 1499–1509, Lisbon, Portugal, 2015.

[43] K. Toutanova and D. Chen, "Observed versus latent features for knowledge base and text inference," in *Proceedings of the 3rd workshop on continuous vector space models and their compositionality*, pp. 57–66, Beijing, China, 2015.

[44] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," 2014, https://arxiv.org/abs/1412.6980.

[45] S. Han, Z. Meng, Z. Li et al., "Optimizing Filter Size in Convolutional Neural Networks for Facial Action Unit Recognition," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5070–5078, Salt Lake City, 2018.

[46] V. Mitra, G. Sivaraman, H. Nam, C. Espy-Wilson, E. Saltzman, and M. Tiede, "Hybrid convolutional neural networks for articulatory and acoustic information based speech recognition," *Speech Communication*, vol. 89, pp. 103–112, 2017.

[47] E. Vargas-Ocampo, E. Roman-Rangel, and J. Hermosillo-Valadez, *Learning Word and Sentence Embeddings Using a Generative Convolutional Network*, Pattern Recognition, pp. 135–144, Springer International Publishing, Cham, 2018.