Received October 2, 2020, accepted October 25, 2020, date of publication November 3, 2020, date of current version November 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3035636

Knowledge Graph Representation Learning With Multi-Scale Capsule-Based Embedding Model Incorporating Entity Descriptions

JINGWEI CHENG, FU ZHANG, AND ZHI YANG
College of Computer Science and Engineering, Northeastern University, Shenyang 110169, China

Corresponding author: Jingwei Cheng (chengjingwei@mail.neu.edu.cn)

This work was supported by the National Natural Science Foundation of China under Grant 61672139.

ABSTRACT A Knowledge Graph (KG) is a directed graph with nodes as entities and edges as relations. KG representation learning (KGRL) aims to embed entities and relations in a KG into continuous low-dimensional vector spaces, so as to simplify the manipulation while preserving the inherent structure of the KG. In this paper, we propose a KG embedding framework, namely MCapsEED (Multi-Scale Capsule-based Embedding Model Incorporating Entity Descriptions). MCapsEED employs a Transformer in combination with a relation attention mechanism to identify the relation-specific part of an entity description and obtain the description representation of an entity. The structured and description representations of an entity are integrated into a synthetic representation. A 3-column matrix with each column a synthetic representation of an element of a triple is fed into a Multi-Scale Capsule-based Embedding model to produce final representations of the head entity, the tail entity and the relation. Experiments show that MCapsEED achieves better performance than state-of-the-art embedding models for the task of link prediction on four benchmark datasets. Our code can be found at https://github.com/1780041410/McapsEED.

INDEX TERMS Knowledge graph representation learning, capsule network, link prediction, knowledge graph embedding.

I. INTRODUCTION

A Knowledge Graph (KG) is a graph of data intended to accumulate and convey knowledge of the real world, whose nodes represent entities of interest and whose edges represent relations between these entities [1]. KGs, such as Freebase [2], Yago [3] and WordNet [4], express precise and effective structured information and have become an important data source for knowledge-driven applications such as information retrieval [5], recommendation systems [6], intelligent question answering systems [7], [8], language representation [9], and semantic similarity searching [10], [11].

KGs evolved from the Semantic Web [12], [13], the essence of which is a directed graph composed of entities connected by relations. Each edge is a triple of the fact (*head entity, relation, tail entity*) (denoted as (h, r, t)). An example triple from Freebase looks like this: `(Hamlet, story_by, William_Shakespeare)`.

The associate editor coordinating the review of this manuscript and approving it for publication was Benyun Shi.

Although effective in representing structured data, two challenges arise when manipulating KGs, the computational complexity problem and the data sparsity problem [14].

To tackle these issues, KG representation learning (KGRL), which aims to map entities and relations in KGs into continuous low-dimensional vector spaces, has been proposed and attracted considerable research interests [15], [16]. Various methods have been proposed on KGRL, which are roughly categorized into two groups: translation-based models and semantic matching models [17], [18]. Among them, the translation-based models are most widely used given their simplicity and effectiveness [19]. However, the inherent shortcoming of translation-based models still exists, as shallow networks cannot adequately extract relevant features of entities and relations.

In view of this, ConvE [20] and ConvKB [21] utilize deep convolution network to model entities and relations. Capsule-based Embedding (CapsE) [22] extends ConvKB by introducing the capsule network [23] after the convolution layer. Multi-Scale Capsule-based Embedding (MCapsE) [24] further extends CapsE with multi-scale convolution kernels in the convolution layer to extract features at different abstract levels.

To further improve the performance of KGRL, there have been substantial works on incorporating additional information, e.g., entity types, relation paths, and entity descriptions. Among them, entity descriptions are deemed to contain the richest semantic information and are widely used in KGRL as an important supplement. Most existing methods [25]–[28] manage to incorporate entity descriptions on the basis of translation-based models. However, the inherent simple structure of translation-based models makes themselves hard to model complex relationships [29], which in turn leads to the inability to cope with entity descriptions well. Therefore, we propose a novel KGRL framework, which takes advantage of both structured information and entity descriptions. We name it as Multi-Scale Capsule-based Embedding model incorporating Entity Descriptions (MCapsEED).

We summarize our main contributions in this paper as follows:

- We propose a novel KG embedding framework MCapsEED which exploits both structured and entity description information. We use a Transformer encoder to obtain entity description representations, and a relation attention mechanism to extract the relation-specific entity description features. We employ a dynamic gate mechanism to integrate the entity description representation and the entity structured representation. We adopt multi-scale capsule network to better capture global semantic features between entities and the relation in a triple.
- We evaluate MCapsEED for the task of link prediction. MCapsEED obtains better performance than existing Capsule-based KGRL models. Comparing with KGRL models incorporating entity descriptions on FB15k and WN18 datasets, MCapsEED performs better on the MR and Hits@10 metrics. MCapsEED shows a better complex relation modeling capability on Hits@10 metrics for N-1 and N-M complex relations.

The rest of the paper is organized as follows. Section II introduces the basic concept of Capsule networks. We describe related works in Section III and explain the proposed method in Section IV. We conduct experiments in Section V and then add summary and discussion in Section VI.

II. CAPSULE NETWORKS

Convolutional Neural Networks (CNNs) are adept in identifying features but are not effective in exploring spatial relationships between features, for example, the relative position, the relative size, and the orientation. In face recognition, if we rotate a part of a human face, CNNs will still consider it to be a human face. CNNs are insensitive to relative orientations and spatial relationships of components. In another word, CNNs only care about whether there are features, and do not care the relative location information of features. Another problem is the pooling layer of CNNs. The design purpose of pooling layers is to solve the translation invariance in an image and reduce the amount of parameters. However, a lot of valuable information will lose through the pooling layer, and the correlation between local features extracted from the convolutional layer and the overall features are ignored. Therefore, CNNs fail to learn the spatial correlation of the features extracted by the convolutional layer.

Capsule Networks (CapsNets) [23] extend CNNs by replacing the scalar output feature detectors with vector-output capsules and max-pooling with routing-by-agreement, whereas retain the characteristics of replicating learned knowledge across space. Capsule networks are able to capture the intrinsic spatial part-whole relationship constituting domain invariant knowledge that bridges the knowledge gap between the source and target domains or tasks, such as cross-domain text classification [30]. A Capsule network contains both convolution layers and capsule layers. A dynamic routing algorithm is used to achieve connections between layers.

A capsule layer consists of capsules that replace CNN neurons to produce vector outputs. A capsule is composed of a group of neurons, each of which represents an attribute feature of a specific class. An example of two capsule layers is shown in Figure 1.

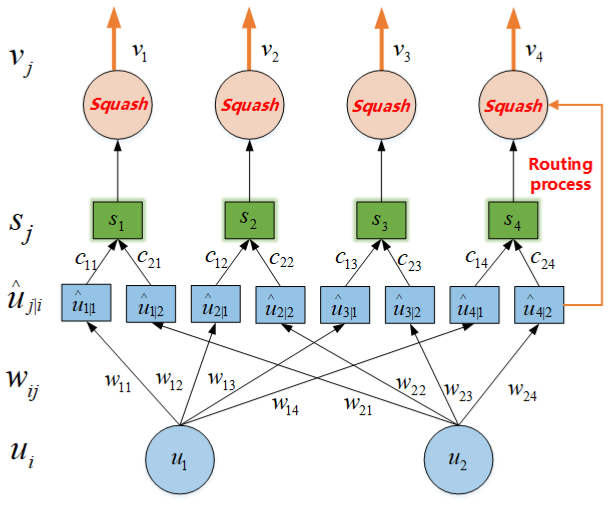


FIGURE 1. An example of two capsule layers.

From bottom to top, the first capsule layer contains two capsules, each of which has a vector output u_i. Vector outputs u_i are multiplied by weight matrices W_{ij} to produce $\hat{u}_{j|i}$, the position output vectors for extracting high-level features from u_i. The position output vectors $\hat{u}_{j|i}$ are multiplied by the coupling coefficients c_{ij}, which is determined by a dynamic routing algorithm, to obtain a weighted sum s_j, the vector inputs to capsules in the second layer. The length of the output of a capsule represents the probability of the existence of the class, which is a real number between 0 and 1. To compress the modulus of the output vector of a capsule to between

0 and 1, a nonlinear squash function is employed to obtain the output vector v_j of the second capsule layer.

$$\hat{u}_{j|i} = W_{ij}u_i \tag{1}$$

$$s_j = \sum_i c_{ij}\hat{u}_{j|i} \tag{2}$$

$$v_j = \frac{||s_j||^2}{1+||s_j||^2}\frac{s_j}{||s_j||} \tag{3}$$

where v_j is the output vector of capsule j.

For further details of Capsule Networks, one can refer the original paper [23] and an excellent illustration [31].

III. RELATED WORK

A. TRANSLATION-BASED MODELS

TransE [15], the earliest translation-based model, was inspired by the fact that the algebraic operation of word vectors in word2vec model is still meaningful. It regards the relation as a translation operation from the head entity to the tail entity. TransE has attracted wide attentions because of its effectiveness and simplicity. However, TransE also has some obvious shortcomings. It fails in modeling complex relations well such as 1-N, N-1 and N-M.

To overcome shortcomings of TransE, TransH [32] assumes entity vectors and relation vectors lie in different hyperplanes. TransH maps the head entity vector and the tail entity vector to the hyperplane where the relation is located to perform a translation operation. TransR [33] introduces relation-specific spaces, rather than hyperplanes. TransR thus maps the head entity vector and tail entity vector to the relation-specific space through a transformation matrix. TransD [34] argues that entities and relations are diverse, so the transformation matrices should be related not only to relations, but also to entities. The only difference between TransD and TransR is that the transformation matrix of TransD model is obtained dynamically from entity vectors and relation vectors. TransSparse [35] uses sparse matrices instead of dense matrices in TransR to solve the problem of heterogeneity of relations, and uses different sparse projection matrices to map head and tail entities to solve the problem of imbalance of relations. These models solve the limitations of TransE in complex relation modeling to a certain extent, and improve the learning of semantic information of entities and relations in KGs.

B. DEEP NEURAL NETWORK MODEL

Translation-based models are mostly shallow neural network models, which are incapable of extracting correlation features between entities and relations.

The application of deep neural network architecture in KGRL dates back to NTN [36]. The embedding layer of NTN embeds entities into vectors. Then, the embeddings of the head and tail entities are combined by a relation-specific tensor and mapped to a non-linear hidden layer. Finally, a score indicates the plausibility of the triple is obtained by a relation-specific linear output layer. The model is trained to maximize this plausibility. SME [37] is another neural network architecture. The embedding layer of SME embeds both entities and relations into vectors, whereas the hidden layer characterizes the interactions of the head entity to the relation, and the tail entity to the relation, respectively. The model is trained to maximize the semantic similarity of these two interactions.

ConvE [20] and ConvKB [21] utilize deep convolution network to model entities and relations. CapsE [22] extends ConvKB by introducing the capsule network [23] after the convolution layer and achieves the state-of-the-art results for the task of KG completion on two benchmark datasets WN18RR and FB15k-237. However, as the convolution layer in the capsule network uses a single window size convolution kernel, the feature map obtained after a convolution operation contains only partial features represented by the head and tail entities and partial interaction features represented by relations. CapS-QuaR [38] uses the capsule network to replace the traditional neural network and uses the quaternion as the input of the model to encode semantics of factual triples. QuaR model defines each relation as a rotation from the head entity to the tail entity in the hyper-complex vector space, which could be used to infer and model diverse relation patterns, including: symmetry/antisymmetry, reversal and combination. To obtain interactive features of larger context, and to obtain more entity features, MCapsE [24] employs multi-scale convolution kernels, i.e., convolution kernels of different windows sizes, in the convolution layer to extract features at different abstract levels. The semantic features of entities and relations are then expressed as continuous vectors through an improved routing process algorithm to form final representations. Experiments on the task of KG completion show that the proposed model is more competitive than state-of-the-art methods, especially in relation classification tasks.

C. INCORPORATING ENTITY DESCRIPTIONS

Early models, such as NTN [36], model entity descriptions separately from KG triples and fail to model interaction between them. Until recently, entity descriptions, as a supplement to the structured information of triples, are incorporated into KGRL models to improve the performance.

DKRL [25] treats entity descriptions as an important component of entity representations, and employs CBOW (Continuous Bag-of-Words) [39] and CNN to encode entity descriptions. DKRL does not consider relation-specific entity description information, and thus fails to integrate entity structured representations and entity description representations effectively. TEKE_H [40] utilizes entity contextual information in KGRL by adopting word2vec and TransH to embed the textual context and entities/relations respectively. The BiLSTM-based joint KGRL model, named Jointly [26], uses BiLSTM to extract entity description information related to relations and achieves significant performance improvement. Jointly has two versions: Jointly(LSTM) and Jointly(A-LSTM), which represent jointly encoding models with LSTM and LSTM+Attention text encoders.

SSP [27] learns entity description representations through topic modelling and restricts the structured representations within the same subspace, but does not fully exploit the semantic relevance of entities and entity descriptions. In [41] entity description information and entity structured information are integrated, under the complete attention mechanism, which consider attentions of the head entity, the tail entity and their relation. An entity is thus supposed to have different representations of corresponding semantics in different triples. In [42], a Multiple Interaction Attention (MIA) mechanism is utilized to model the interactions between the head entity description, the head entity name, the relation name, and candidate tail entity descriptions, to form enriched representations. Besides triple and text descriptions, TDN [43] additionally integrates network structure of a KG in KGRL.

In order to extract the entity description features related to the relations more efficiently, we adopt Transformer in combination of the relation attention mechanism. We use the dynamic gate mechanism to integrate entity description representations and entity structured representations to improve the effect of KGRL.

IV. MULTI-SCALE CAPSULE EMBEDDING MODEL INCORPORATING ENTITY DESCRIPTIONS

The problem of sparseness in KGs still exists. For entities occurring in a small number of triples, when learning their representations based on only triples, the obtained vector representations are difficult to contain rich semantic information. In fact, almost every entity in a KG has a description on Wikipedia that describes its specific meaning and contains rich semantic information. Therefore, incorporating entity descriptions is conducive to enhancing the effect of knowledge representation learning and helps alleviating the sparseness of entity representation in a KG.

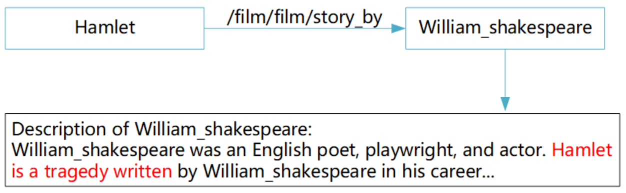
Some parts of the entity description are important for an entity in a given relation, whereas others are irrelevant. Therefore, how to accurately learn the entity description information related to the relation and ignore the irrelevant information is of great importance, which is also a key component of this study. For example, in Figure 2, the red text shows the description of an entity `William_Shakespeare` related to the relation `/film/film/story_by` in FreeBase.

A. ACQUIRING AND PREPROCESSING ENTITY DESCRIPTIONS

Obtaining entity descriptions is not a hard nut to crack. In KGs such as Freebase and WordNet, most entities have related description information. For a small number of entities that have no description, we align them to Wikipedia resources to obtain the entity description information and store them in json format.

We preprocess entity descriptions by removing non-textual symbols and special characters, converting uppercase characters to lowercase, and performing word segmentation. After preprocessing, in Freebase, the average length of entity descriptions is 69, and the maximal length of entity descriptions is 343. In WordNet, the average length and the maximal length are 13 and 96 respectively.

The preprocessed result is denoted as $desc = \{w_1, w_2, \ldots, w_n\}$ and is fed into an entity-description-aware KGRL model.



Description of William_shakespeare:
William_shakespeare was an English poet, playwright, and actor. Hamlet is a tragedy written by William_shakespeare in his career...

FIGURE 2. An example of an entity description in Freebase.

B. FRAMEWORK

As is shown in Figure 3, the framework of Multi-Scale Capsule Embedding model incorporating Entity Descriptions (MCapsEED) consists of three modules: Entity Description Embedding Learning, Integration of Structured and Description Information, and Multi-Scale Capsule-based Embedding (MCapsE) Learning.

The preprocessed entity descriptions are fed into the framework as the input of the Entity Description Encoder, where Transformer in combination with relation attention mechanism is used to encode head and tail entity descriptions into vector representations h_d and t_d. Through dynamic gate mechanism, h_d and t_d are integrated with structured representations of head and tail entities from TransE model, h_s and t_s, to obtain the synthetic representations of the head and tail entities, v_h and v_t. MCapsE perform representation learning on v_h and v_t, and the structured representation v_r of the relation to obtain the final representations of the head entity, the tail entity and the relation. We will detail the three modules in the following.

C. ENTITY DESCRIPTION EMBEDDING LEARNING

Entity descriptions contain abundant semantic features of related entities. As we will show in Section V, efficiently and accurately extracting important semantic features entailed in entity descriptions will substantially improve the performance of a KGRL model.

Given a triple (h, r, t), we use Transformer encoder and relation Attention mechanism to obtain adequate semantic information related to the relation r contained in the entity description of h or t. As the processes of obtaining representations of descriptions for the head entity and the tail entity are the same, we use the head entity as an example. The model for entity description embedding learning is shown in Figure 4, which consists of three layers: input layer, Transformer Encoder layer, and relational attention layer.

1) INPUT LAYER

In the input layer, word embeddings and position embeddings of an entity description are concatenated to form the sentence embedding, which serve as model input information. Due to

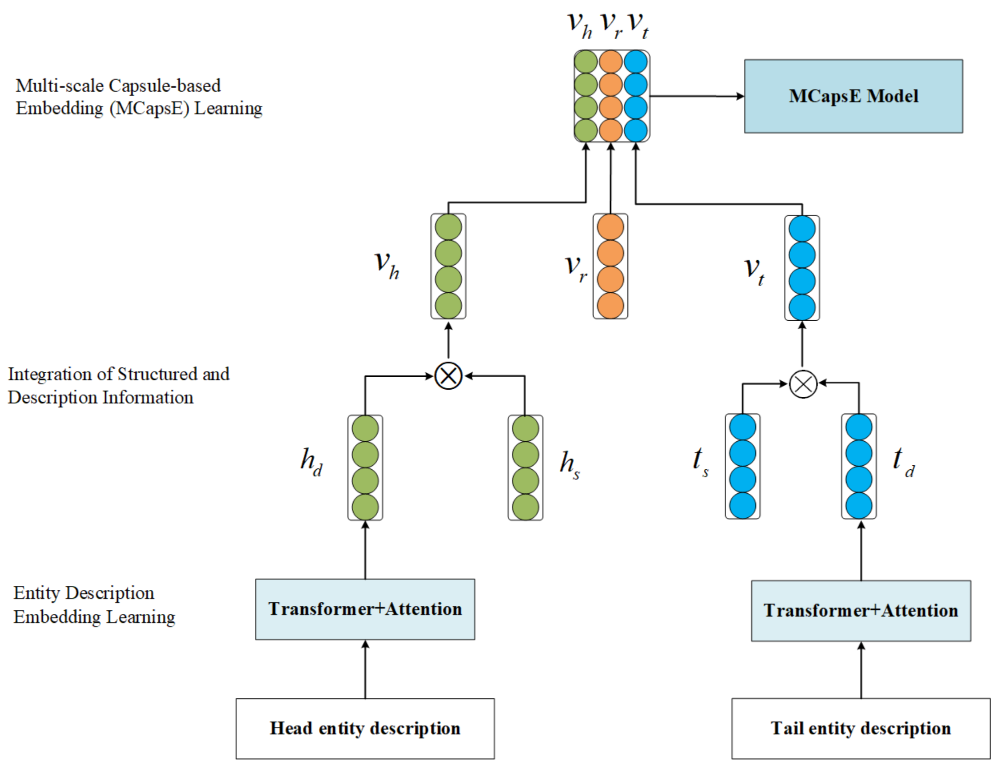


FIGURE 3. **Framework of multi-scale capsule embedding model incorporating entity descriptions.**

its simplicity and effectiveness [44], we choose Skip-gram to train the word embeddings from large amounts of entity descriptions. We set the size of skip window to 5, and the dimension of word embeddings $d = 100$. After training, we obtain the word embedding matrix $WordVec \in R^{k\times d}$, where k is the vocabulary size of entity descriptions.

Through a lookup in the obtained word embedding matrix, we can obtain an embedding x_i for each word w_i in an entity description $desc = \{w_1, w_2, \ldots, w_n\}$, and in turn obtain a word embedding matrix $X = (x_1, x_2, \ldots, x_n\}$ of the entity description, where $x_i \in R^{d\times 1}$ is the embedding of the i-th word of the entity description, and n is the length of the entity description.

To capture sequence ordering of entity descriptions, we make use of position embeddings obtained by looking up a randomly initialized position embedding matrix $PosVec \in R^{k\times d}$, which is updated during training. After each position index is converted to a position embedding, we obtain the position embedding matrix $P = (p_1, p_2, \ldots, p_n)$, where $p_i \in R^{d\times 1}$ is the position embedding of the i-th word in the description, and n is the length of the entity description.

We concatenate the word embedding and the position embedding to obtain the output vector $S = (s_1, s_2, \ldots, s_n)$ of input layer, where $s_i \in R^{2d\times 1}$ is the concatenation of w_i and p_i.

2) TRANSFORMER ENCODER LAYER

To obtain more semantic features in a entity description, and to learn the dependency relationship of words in the description, we adopt a Transformer Encoder with the multi-head self-attention mechanism. The encoder is composed of a stack of $N = 6$ identical layers. Each layer has two sub-layers, a multi-head self-attention layer and a simple position-wise fully connected feed-forward network. We employ a residual connection around each of the two sub-layers, followed by layer normalization.

We denote Query/Key/Value vectors as Q, K and V. For a given entity description embedding S, the self attention mechanism demands $Q = K = V = S$. Here we adopt dot product, as is shown in (4), for simplicity and more complicated aggregation strategies are left for future work.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d}})V \tag{4}$$

where d is the dimension of Q and K.

For each head, we have a set of randomly initialized Query/Key/Value weight matrices, through which we map Q, K and V to different matrices.

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{5}$$

After calculating attentions separately on multiple attention heads, we concatenate and condense them into a single matrix.

$$\begin{aligned} Y &= MultiHead(Q, K, V) \\ &= Concat(head_1, head_2, \ldots, head_n)W^o \end{aligned} \tag{6}$$

Then, we use two layers of residual connection followed by a layer-normalization step (see (7) and (9)), and two linear

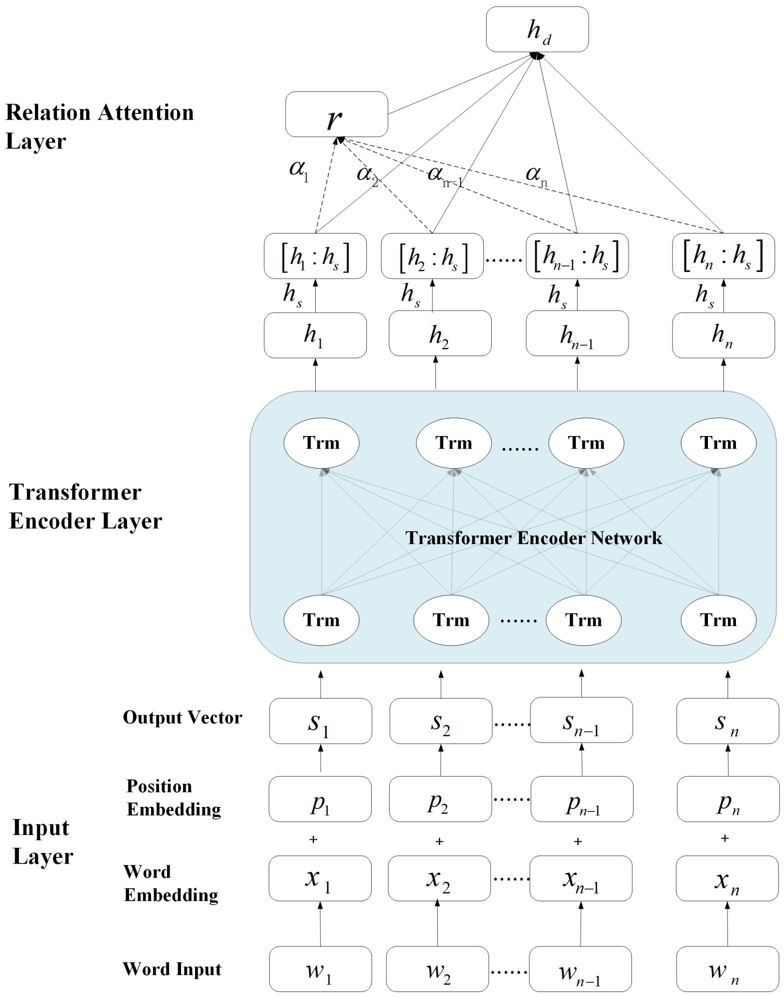


FIGURE 4. Entity description embedding learning.

layers with a ReLU activation between them (see (8)).

$$L = LayerNorm(S + Y) \tag{7}$$
$$FFN(L) = \mathrm{ReLU}(LW_1 + b_1)W_2 + b_2 \tag{8}$$
$$H = LayerNorm(L + FFN(L)) \tag{9}$$

where W_1 and W_2 are weight parameter matrices, and b_1 and b_2 are bias vectors.

Finally, the output of the Transformer encoder layer is the representation of each word with its contextual semantic information $H = (h_1, h_2, \ldots, h_n)$, which serves as the input of the relation attention layer.

3) RELATION ATTENTION LAYER

To obtain a global vector representation of the entity description, a simple and direct way is to average vector representations of words in the entity description. However, this approach treats each word in the entity description indiscriminately, without considering the importance of words related to the relation in a triple. We thus employ a relation attention mechanism to calculate the weight of each word in the entity description, and come up with the global representation of the entity description as the weighted sum of representations of words in the entity description.

To identify, in a entity description, which words are closely related to the entity and the relation in a triple, we calculate the weight of each word in the entity description using a simple fully connected neural network, the input of which are the head entity representation h_s and the relation representation h_r pre-trained by TransE model, and the contextual feature representation h_i of each word. We calculate the weight in (10).

$$\begin{aligned} f_i &= W_a[h_i; h_s] + Vh_r \\ \alpha_i &= \frac{\exp(f_i)}{\sum_{j=1}^{N} \exp(f_j)} \\ h_d &= \sum_{i=1}^{N} \alpha_i h_i \end{aligned} \tag{10}$$

where W_a is the weight matrix, $V \in R^{d\times 1}$ is the parameter vector, and h_d is the representation of the head entity

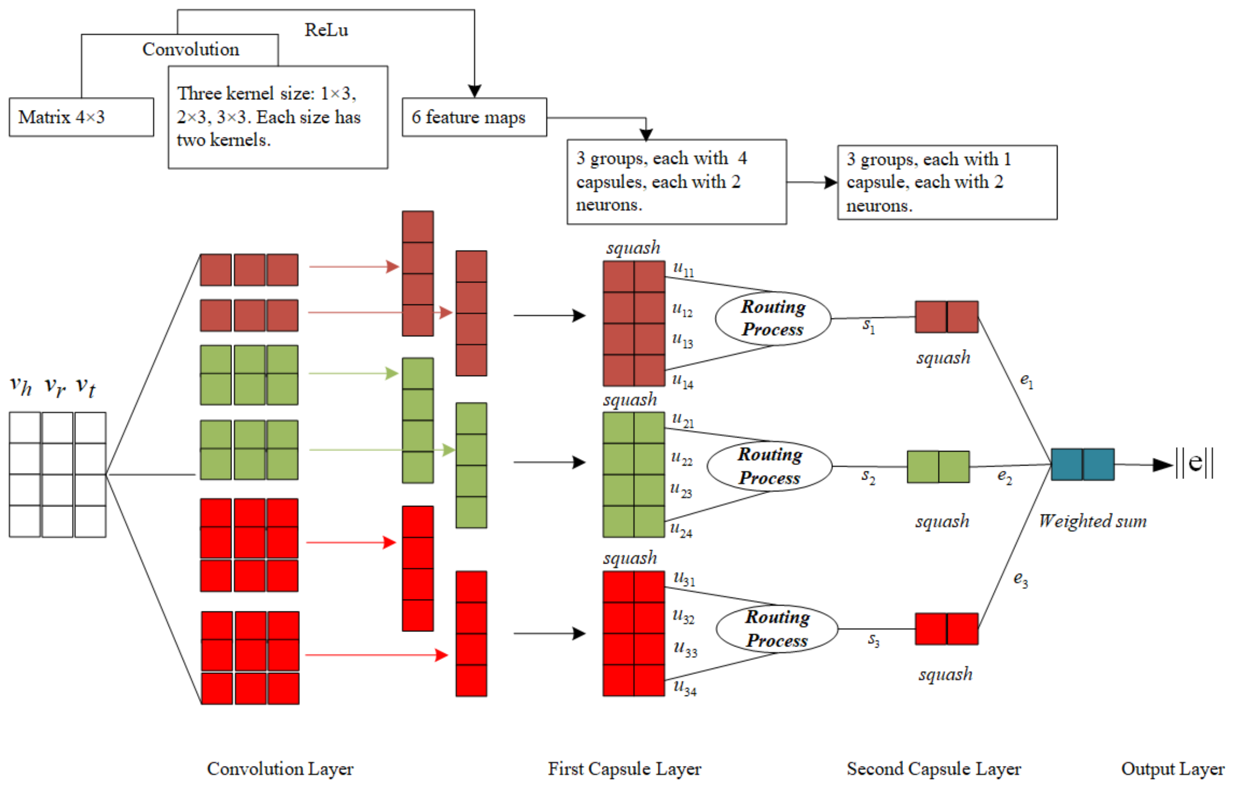

Matrix 4×3
Convolution
Three kernel size: 1×3, 2×3, 3×3. Each size has two kernels.
ReLu
6 feature maps
3 groups, each with 4 capsules, each with 2 neurons.
3 groups, each with 1 capsule, each with 2 neurons.
v_h v_r v_t
squash
u_{11} u_{12} u_{13} u_{14}
u_{21} u_{22} u_{23} u_{24}
u_{31} u_{32} u_{33} u_{34}
Routing Process
s_1 s_2 s_3
e_1 e_2 e_3
Weighted sum
$\|e\|$
Convolution Layer
First Capsule Layer
Second Capsule Layer
Output Layer

FIGURE 5. Framework of MCapsE with $k = 4$, $N = 2$, and $d = 2$.

description. In the same way, we can obtain the representation t_d of the tail entity description.

D. INTEGRATION OF ENTITY DESCRIPTION REPRESENTATION AND STRUCTURED REPRESENTATION

Entity descriptions contain rich semantic information, which can be used as a supplement to structured triple information. To make full use of the entity description information, we integrate representations of the head and tail entity descriptions, and structured representations of the head and tail entities, to obtain the synthetic representations of the head and tail entities. We explore two different integration methods, the direct concatenation and the dynamic gate mechanism.

1) DIRECT CONCATENATION

The representation of head entity description and the structured representation of head entity are concatenated on their last dimension to obtain the intermediate representation, which is fed into a fully connected layer followed by a ReLU activation to output their integration result. In the same way, the integration result for tail entity is obtained. The calculations for the head and the tail entity are shown in (11) and (12).

$$v_h = \mathrm{ReLU}(W_e[h_d; h_s] + b_e) \tag{11}$$
$$v_t = \mathrm{ReLU}(W_e[t_d; t_s] + b_e) \tag{12}$$

where W_e is a shared parameter for both head and tail entities.

2) DYNAMIC GATE MECHANISM

First, we use the head entity description representation and the head entity structured representation, h_d and h_s, to calculate a weight gate vector g, and then use g to integrate h_d and h_s to form the synthetic representation v_h. The calculation process for the synthetic representation of the tail entity v_t is in a same way. The calculations for the head and tail entity are shown in (13) and (14).

$$\begin{aligned} g &= \mathrm{Softmax}(W_1[h_d; h_s] + b_1) \\ v_h &= g \odot h_d + (1-g) \odot h_s \end{aligned} \tag{13}$$

$$\begin{aligned} g &= \mathrm{Softmax}(W_1[t_d; h_s] + b_1) \\ v_t &= g \odot t_d + (1-g) \odot t_s \end{aligned} \tag{14}$$

where $\odot$ is called Hadamard product or element-wise product, and W_1 and b_1 are shared parameters.

Comparative experiments and the result analysis of the two integration methods will be given in Section V.

E. MULTI-SCALE CAPSULE-BASED EMBEDDING MODEL

Figure 5 shows the framework of MCapsE. The function of each layer of the model is elaborated as follows.

1) EMBEDDING LAYER

We treat each embedding triple (v_h, v_r, v_t) as a matrix $A = [v_h, v_r, v_t] \in R^{k\times 3}$, where $A_{i,:} \in R^{1\times 3}$ is the i-th row of A.

2) CONVOLUTION LAYER

The input of the convolution layer is the embedding matrix A. We use three different window sizes $j \times 3$, where $j \in \{1, 2, 3\}$. For each window size, we employ N convolution kernels $\omega_j \in R^{j\times 3}$. As is shown in (15), a convolutional operation is executed on each row of the matrix A by using N convolution kernels to produce feature maps. We thus have $3N$ k-dimensional feature maps, for which each feature map can capture one single characteristic among entries at the same dimension. For each convolution kernel, we have

$$c_{j,i} = \text{ReLU}\left(\omega_j \cdot A_{i,:} + b\right) \tag{15}$$

$$q_j = \left[c_{j1}, c_{j2}, \ldots, c_{jk}\right] \in R^k \tag{16}$$

$$q = [q_1, q_2, q_3] \tag{17}$$

where $\cdot$ is the dot product operation, $b \in R$ is the offset vector.

Feature maps generated by convolution kernels of the same size form a feature map list. We thus have three feature map lists as the input of the first capsule layer.

3) CAPSULE LAYERS

We use two capsule layers in MCapsE. In the first layer, we construct k capsules for each feature map list. We encapsulate features in the same dimension in the feature map list into a same capsule to capture features at different positions in the triple embedding. For each capsule, we thus have a corresponding vector $u_{ji} \in R^{N\times 1}$. Vector u_{ji} of each capsule $i \in \{1, 2, \ldots, k\}$ are multiplied by weight matrix $W_{ji} \in R^{d\times N}$ to obtain vector $\hat{u}_{ji} \in R^{d\times 1}$. Vectors $\hat{u}_{ji}$ are weighted summed to obtain an input vector $s_j \in R^{d\times 1}$ of the second capsule layer. A nonlinear compression function is executed on s_j to generate a vector output $e_j \in R^{d\times 1}$. Vectors e_1, e_2, e_3 are weighted summed to obtain e, the length of which represents the score of the triple. The process is specified in (18).

$$e_j = squash\left(s_j\right), s_j = \sum_i c_i \hat{u}_{ji}, \hat{u}_{ji} = W_{ji} u_{ji}$$

$$e = \sum_{j\in\{1,2,3\}} e_j \tag{18}$$

In (19), we change the number in the denominator of the squash function from 1 to 0.5, so that the vector features are enlarged before the modulus length reaches 0, which is beneficial to capture the correlation between features.

$$squash\left(s_i\right) = \frac{\|s_i\|^2}{0.5 + \|s_i\|^2} \frac{s_i}{\|s_i\|} \tag{19}$$

F. MODEL TRAINING

We use the Adam Optimizer [45] to train the proposed KGRL model by minimizing the cross entropy loss function in (20).

$$\mathcal{L} = - \sum_{(h,r,t)\in\{S\cup S'\}} \left(t_{(h,r,t)} \log f(h, r, t) + \left(1 - t_{(h,r,t)}\right) \log f(h, r, t)\right) \tag{20}$$

where the scoring function f in defined in (21),

$$f(h, r, t) = \|\texttt{MCapsE}\left(g\left([v_h, v_r, v_t] * \Omega\right)\right)\| \tag{21}$$

TABLE 1. Hyper-parameters for different datasets.

Hyperparameters	FB15k-237	WN18RR	FB15k	WN18
Learning rate	1e-5	5e-6	1e-5	1e-5
Batch size	128	256	128	256
Regularization factor	1e-5	1.2e-4	1.2e-5	1e-5
Num of Conv kernels	400	400	400	400
1st kernel size	3×3	3×3	1×3	1×3
2nd kernel size	4×3	4×3	2×3	2×3
3rd kernel size	5×3	5×3	3×3	3×3
Dropout rate	0.2	0.2	0.2	0.2
epoch	400	300	300	200
Relation embedding dim	100	100	100	100
Entity embedding dim	100	100	100	100
Word embedding dim	100	100	100	100
Number of capsules	100	100	100	100
Dynamic routing iterations	3	2	5	7

where MCapsE denotes a MCapsE network operator, Ω is the shared parameter in the convolutional layer, and $*$ represents the convolution operator.

$t_{(h,r,t)}$ is defined as $t_{(h,r,t)} = \begin{cases} 1, & \text{if } (h, r, t) \in S \\ -1, & \text{if } (h, r, t) \in S' \end{cases}$, where S is the positive triple set, and S' is the negative triple set.

In addition, as too many network layers in the Transformer Encoder may result in a shift in data distribution. To prevent this phenomena from occurring and accelerate convergence and improve the generalization ability of the model, we add a Batch Normalization layer [46] and a SpatialDropout [47] before and after the Transformer Encoder layer. As we will show in Section V, these methods significantly improve the performance of the representation model.

V. EXPERIMENT AND ANALYSIS

A. EXPERIMENTAL DESIGN

The experiment contain two parts:

(1) Comparison of MCapsEED with existing Capsule-based KGRL models, CapsE and MCapsE, to verify whether incorporating entity description information improves the performance of KGRL models. The experimental datasets are consistent with that of in MCapsE and CapsE, namely FB15k-237 and WN18RR.

(2) Comparison with existing KGRL models incorporating entity description information to prove the effectiveness and generalization of the proposed model. As existing models of this kind use FB15k and WN18 as benchmarks, we compare with them on these two datasets.

In the process of implementing of MCapsEED, we conduct comparative analysis experiments on whether the entity description feature extraction, dynamic gate mechanism, and multi-scale capsule network representation method have improved the model. To obtain the best experimental hyperparameters, we use a Grid Search strategy. As we use four different datasets for comparative analysis, there are four optimal hyperparameter lists, which is shown in Table 1.

B. EXPERIMENTAL DATA AND EVALUATION METRICS

Task We evaluate MCapsEED on the task of link prediction, the goal of which is to predict a missing entity given a relation and another entity in a triple.

TABLE 2. Statistics of datasets.

Dataset	Relations	Entities	Train Set	Val Set	Test Set
FB15k-237	237	14,541	272,115	17,535	20,466
WN18RR	11	40,943	86,835	3,034	3,134
FB15k	1,345	14,951	483,142	50,000	59,071
WN18	18	40,493	141,442	5,000	5,000

TABLE 3. Comparison of MCapsEED with CapsE and MCapsE.

	WN18RR			FB15K-237		
Model	MR	MRR	Hits@10	MR	MRR	Hits@10
CapsE [22]	719	0.415	56.0	303	0.523	59.3
MCapsE [24]	710	0.402	57.2	317	**0.532**	60.5
MCapsEED	**659**	**0.425**	**58.6**	**296**	0.531	**62.3**

Datasets We use commonly used datasets, FB15k-237, WN18RR, FB15k, and WN18. Table 2 lists the detailed information of four datasets.

Metrics After representations of entities and relationships in a KG is learned, the link prediction task is transformed into a ranking procedure. Taking the task of predicting the head entity as an example, i.e., $(?, r, t)$, each entity h in the KG is a candidate answer. For each candidate triple (h, r, t), the score is calculated by a scoring function, e.g. (21) in our case of MCapsEED. Sorting these scores in descending order will produce a ranked list of candidate answers.

For evaluation, it is common to record ranks of correct answers in such a ranked list to see if correct answers rank before incorrect ones. Various evaluation metrics have been designed based on such ranks. The evaluation metrics used in this paper are, under the "filter" mode, the Mean Rank (MR, the average of predicted ranks), the Mean Reciprocal Rank (MRR, the average of reciprocal ranks), and Hits@n (the proportion of ranks no larger than n). Lower MR, higher MRR, and higher Hits@n indicate better performance. The "filter" mode [15] means not taking any negative triples that appear in the KG into accounts. To facilitate comparison, we employ the common Bernoulli strategy [32] used in CapsE and MCapsE when sampling negative triples.

C. ANALYSIS OF RESULTS

1) COMPARISON WITH CapsE AND MCapsE

We first compare with MCapsE and CapsE models that do not consider entity description information, the result of which is listed in Table 3. It can be drawn that, by incorporating entity description information, the performance of the MCapsEED is significantly better than MCapsE and CapsE, especially in the metrics of Hits@10, which indicates that MCapsEED further improve the discrimination of entity representation. On the other hand, it can also be derived from the results on MR metrics that, incorporating entity description information makes the correct entities rank higher among the candidate entities, which greatly reduces the sparsity of entity representations.

We also compare the performance of these three models on relations of different categories, which is shown in Table 4. We divide the relations into four categories, i.e., 1-1, 1-N, N-1, and N-M.

TABLE 4. Model performance on different category relations in FB15k-237 dataset.

	Prediction Head (Hits@10)				Prediction Tail (Hits@10)			
Model	1-1	1-N	N-1	N-M	1-1	1-N	N-1	N-M
CapsE [22]	47.4	47.72	59.78	57.28	**0.45**	0.17	0.69	0.52
MCapsE [24]	48.2	47.23	59.95	59.63	0.43	**0.18**	0.73	0.56
MCapsEED	**48.5**	**48.31**	**60.2**	**62.6**	0.44	0.172	**0.75**	**0.58**

TABLE 5. Model performance of different integration methods and optimization strategies on FB15k-237 dataset.

Model	Hits@10
MCapsEED (concat)	60.2
MCapsEED (gate)	**61.1**
MCapsEED (gate + att)	61.4
MCapsEED (gate + att + bn)	61.7
MCapsEED (gate + att + bn + sd)	**62.3**

It can be seen from Table 4 that in terms of different relation types, MCapsEED shows comparable performance with MCapsE and CapsE models. MCapsEED has a good effect on Hits@10 metrics, especially for N-1 and N-M complex relations. Therefore, MCapsEED can make full use of entity descriptions as an important supplement of structured representations, so as to better deal with complex relation modeling.

2) DIFFERENT INTEGRATION METHODS AND OPTIMIZATION STRATEGIES

In addition, we explore the performance of MCapsEED under different integration methods and optimization strategies, the result of which is shown in Table 5.

As is introduced in Section IV-D, we adopt two different ways of integrating the entity description representation and the entity structured representation, where `concat` represents direct concatenation of two representations, whereas `gate` represents the dynamic gate mechanism. The dynamic gate mechanism improves the @Hits@10 metrics by nearly one percent, which proves its effectiveness.

MCapsEED consists of three modules: Entity Description Embedding Learning, Integration of Structured and Description Information, and MCapsE Learning. In the Entity Description Embedding Learning module, we use a Transformer encoder to obtain entity description representations, and a relation attention mechanism to extract relation-specific entity description features. We add a Batch-Normalization layer and a SpatialDropout layer before and after the Transformer Encoder layer to prevent the shift of data distribution. In the Integration of Structured and Description Information module, we employ a dynamic gate mechanism to integrate the entity description representation and the entity structured representation. In the MCapsE Learning module, we adopt multi-scale capsule network to better capture global semantic features of between entities and the relation in a triple.

TABLE 6. **Comparison with knowledge representation models incorporating entity descriptions.**

	WN18		FB15k	
Model	MR	Hits@10	MR	Hits@10
TransE (Baseline) [15]	251	89.2	125	47.1
Jointly(LSTM) [26]	**95**	91.6	90	69.7
Jointly(A-LSTM) [26]	123	90.9	73	75.5
DKRL [25]	-	-	91	67.4
TEKE_H [40]	114	92.9	108	73.0
SSP [27]	156	93.2	82	79.0
MCapsEED	105	**96**	**69**	**88.1**

We conduct ablation studies on these three modules. To fully exploit the entity description information related to the relation, MCapsEED using dynamic gate mechanism is augmented with a relation attention mechanism, denoted by `att`, which also improves the knowledge representation effect by 0.3%. To prevent the risk of overfitting and enhance the robustness of the model, Batch-Normalization (denoted by `bn`) is introduced and results in a performance improvement of additional 0.3%. When integrating additionally SpatialDropout (denoted by `sd`), we achieve an overall performance improvement of 1.2%.

3) COMPARISON WITH EXISTING KGRL MODELS INCORPORATING ENTITY DESCRIPTIONS

We compare MCapsEED with existing KGRL models incorporating entity descriptions, including Jointly(LSTM), Jointly(A-LSTM), DKRL, TEKE_H, SSP, and use TransE model as baseline.

It can be seen from Table 6 that MCapsEED performs better than existing KGRL models incorporating entity descriptions on the MR and Hits@10 metrics. The main reason is that MCapsEED uses the relation attention mechanism to extract relation-specific features in the entity description. Furthermore, MCapsEED integrates the feature representation of entity descriptions and the structured representation learned from triples through a dynamic gate mechanism, which greatly improves the performance. On WN18, MCapsEED is slightly worse than Jointly(LSTM), which may caused by the limited number of relations in WN18. The attention mechanism thus has no obvious advantage. On FB15K, MCapsEED achieves the best performance and is significantly higher than other models.

VI. CONCLUSION

We propose a KGRL framework which consists of three modules: Entity Description Embedding Learning, Integration of Structured and Description Information, and Multi-Scale Capsule-based Embedding Learning. We use a Transformer encoder to obtain the entity description representations, and the relation attention mechanism to extract the relation-specific entity description features. We employ the dynamic gate mechanism to integrate the entity description representation and the entity structured representation. We adopt multi-scale capsule network to better capture global semantic features between entities and the relation in a triple.

Experiment results are consistent with our design intention. Incorporating entity descriptions improves the performance, especially in the metrics of Hits@10, which indicates that MCapsEED further improve the discrimination of entity representation and greatly reduces the sparsity of entity representations. MCapsEED shows a better complex relation modeling capability on Hits@10 metrics for N-1 and N-M complex relations. MCapsEED also performs better than existing KGRL models incorporating entity descriptions on the MR and Hits@10 metrics.

In the future, we will consider to extend our method to uncertain KGs, i.e., KGs that model the inherent uncertainty of relations facts with a confidence score. The representation of uncertain knowledge will provide more natural characterization of the knowledge and benefit downstream applications such as question answering and semantic search [48]. Another research direction concerns the security of MCapsEED and other KGRL models, i.e., designing adversarial attacks against them, improving their adversarial robustness, and evaluating the effect of proposed improvement on their interpretability [49].

REFERENCES

[1] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J. E. L. Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A.-C. Ngonga Ngomo, S. M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, and A. Zimmermann, "Knowledge graphs," 2020, *arXiv:2003.02320*. [Online]. Available: http://arxiv.org/abs/2003.02320

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data (SIGMOD)*, 2008, pp. 1247–1250.

[3] F. M. Suchanek, G. Kasneci, and G. Weikum, "Yago: A core of semantic knowledge," in *Proc. 16th Int. Conf. World Wide Web (WWW)*, 2007, pp. 697–706.

[4] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[5] C. Xiong, R. Power, and J. Callan, "Explicit semantic ranking for academic search via knowledge graph embedding," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 1271–1279.

[6] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 353–362.

[7] Y. Zhang, K. Liu, S. He, G. Ji, Z. Liu, H. Wu, and J. Zhao, "Question answering over knowledge base with neural attention combining global knowledge information," 2016, *arXiv:1606.00979*. [Online]. Available: http://arxiv.org/abs/1606.00979

[8] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," 2014, *arXiv:1412.6575*. [Online]. Available: http://arxiv.org/abs/1412.6575

[9] W. Liu, P. Zhou, Z. Zhao, Z. Wang, Q. Ju, H. Deng, and P. Wang, "K-BERT: Enabling language representation with knowledge graph," in *Proc. AAAI*, 2020, pp. 2901–2908.

[10] Z. Liu, V. W. Zheng, Z. Zhao, F. Zhu, K. C.-C. Chang, M. Wu, and J. Ying, "Semantic proximity search on heterogeneous graph by proximity embedding," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 154–160.

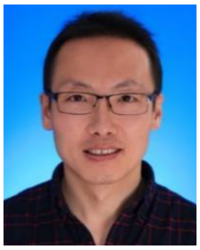
[11] Z. Liu, V. W. Zheng, Z. Zhao, F. Zhu, K. C. C. Chang, M. Wu, and J. Ying, "Distance-sware dag embedding for proximity search on heterogeneous graphs," in *Proc. 32nd AAAI Conf. Artif. Intell. (AAAI)*. Palo Alto, CA, USA: AAAI Press, 2018, pp. 2355–2362.

[12] A. Maedche and S. Staab, "Ontology learning for the semantic Web," *IEEE Intell. Syst.*, vol. 16, no. 2, pp. 72–79, Mar. 2001.

[13] J. Cardoso and A. Sheth, "The semantic Web and its applications," *Semantic Web Beyond*, vol. 284, pp. 3–33, Oct. 2006.

[14] Y. Dai, S. Wang, N. N. Xiong, and W. Guo, "A survey on knowledge graph embedding: Approaches, applications and benchmarks," *Electronics*, vol. 9, no. 5, p. 750, May 2020.
[15] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.
[16] D. Quoc Nguyen, "A survey of embedding models of entities and relationships for knowledge graph completion," 2017, *arXiv:1703.08098*. [Online]. Available: http://arxiv.org/abs/1703.08098
[17] M. Nickel, V. Tresp, and H.-P. Kriegel, "A three-way model for collective learning on multi-relational data," in *Proc. 28th Int. Conf. Int. Conf. Mach. Learn.*, 2011, pp. 809–816.
[18] P. Chen, Y. Wang, Q. Yu, Y. Fan, and R. Feng, "Hamming distance encoding multihop relation knowledge graph completion," *IEEE Access*, vol. 8, pp. 117146–117158, 2020.
[19] R. Xie, Z. Liu, and M. Sun, "Representation learning of knowledge graphs with hierarchical types," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 2965–2971.
[20] T. Dettmers, P. Minervini, P. Stenetorp, and S. Riedel, "Convolutional 2D knowledge graph embeddings," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–9.
[21] D. Q. Nguyen, T. D. Nguyen, D. Q. Nguyen, and D. Phung, "A novel embedding model for knowledge base completion based on convolutional neural network," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 2, 2018, pp. 1–7.
[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 *Proc. Conf. North*, 2019, pp. 2180–2189.
[23] S. Sabour, N. Frosst, and G. E. Hinton, "Dynamic routing between capsules," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3856–3866.
[24] J. Cheng, Z. Yang, J. Dang, C. Pan, and F. Zhang, "Representation learning of knowledge graphs with multi-scale capsule network," in *Proc. Int. Conf. Intell. Data Eng. Automated Learn.* Cham, Switzerland: Springer, 2019, pp. 282–290.
[25] R. Xie, Z. Liu, J. Jia, H. Luan, and M. Sun, "Representation learning of knowledge graphs with entity descriptions," in *Proc. 13th AAAI Conf. Artif. Intell.* Palo Alto, CA, USA: AAAI Press, 2016, pp. 2659–2665.
[26] J. Xu, X. Qiu, K. Chen, and X. Huang, "Knowledge graph representation with jointly structural and textual encoding," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 1318–1324.
[27] H. Xiao, M. Huang, L. Meng, and X. Zhu, "SSP: semantic space projection for knowledge graph embedding with text descriptions," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2017, pp. 1–7.
[28] B. Shi and T. Weninger, "Open-world knowledge graph completion," in in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.
[29] H. Gao, J. Shi, G. Qi, and M. Wang, "Triple context-based knowledge graph embedding," *IEEE Access*, vol. 6, pp. 58978–58989, 2018.
[30] M. Yang, W. Zhao, L. Chen, Q. Qu, Z. Zhao, and Y. Shen, "Investigating the transferring capability of capsule networks for text classification," *Neural Netw.*, vol. 118, pp. 247–261, Oct. 2019.
[31] J. Hui. *Understanding Dynamic Routing Between Capsules (Capsule Networks)*. Accessed: 2017. [Online]. Available: https://jhui.github.io/2017/11/03/Dynamic-Routing-Between-Capsules/
[32] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.
[33] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.
[34] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics, 7th Int. Joint Conf. Natural Lang. Process.* , vol. 1, 2015, pp. 687–696.
[35] G. Ji, K. Liu, S. He, and J. Zhao, "Knowledge graph completion with adaptive sparse transfer matrix," in *Proc. 13th AAAI Conf. Artif. Intell.*, 2016, pp. 1–7.
[36] R. Socher, D. Chen, C. D. Manning, and A. Ng, "Reasoning with neural tensor networks for knowledge base completion," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 926–934.
[37] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Mach. Learn.*, vol. 94, no. 2, pp. 233–259, Feb. 2014.
[38] H. Chen, W. Wang, G. Li, and Y. Shi, "A quaternion-embedded capsule network model for knowledge graph completion," *IEEE Access*, vol. 8, pp. 100890–100904, 2020.
[39] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*. [Online]. Available: http://arxiv.org/abs/1301.3781
[40] Z. Wang and J. Li, "Text-enhanced representation learning for knowledge graph," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, 2016, pp. 1293–1299.
[41] M. Zhao, Y. Zhao, and B. Xu, "Knowledge graph completion via complete attention between knowledge graph and entity descriptions," in *Proc. 3rd Int. Conf. Comput. Sci. Appl. Eng. (CSAE)*, 2019, pp. 1–6.
[42] C. Fu, Z. Li, Q. Yang, Z. Chen, J. Fang, P. Zhao, and J. Xu, "Multiple interaction attention model for open-world knowledge graph completion," in *Proc. Int. Conf. Web Inf. Syst. Eng.* Cham, Switzerland: Springer, 2019, pp. 630–644.
[43] X. Kang, H. Yao, Q. Li, X. Li, C. Liu, and L. Dong, "TDN: An integrated representation learning model of knowledge graphs," *IEEE Access*, vol. 7, pp. 55199–55205, 2019.
[44] X. Chen, Z. Liu, and M. Sun, "A unified model for word sense representation and disambiguation," in *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, 2014, pp. 1025–1035.
[45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980
[46] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[47] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 648–656.
[48] X. Chen, M. Chen, W. Shi, Y. Sun, and C. Zaniolo, "Embedding uncertain knowledge graphs," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 3363–3370.
[49] P. Bhardwaj, "Towards adversarially robust knowledge graph embeddings," in *Proc. AAAI*, 2020, pp. 13712–13713.

JINGWEI CHENG received the Ph.D. degree from Northeastern University, China, in 2011. He is currently with the School of Computer Science and Engineering, Northeastern University. He has authored more than 40 refereed international journals and conference papers, such as WI, DEXA, and IDEAL. He has also authored one monograph published by Springer. His current research interests include knowledge graphs, natural language processing, description logic, RDF data management, and ontologies.



FU ZHANG received the Ph.D. degree from Northeastern University, China, in 2011. He is currently an Associate Professor and a Ph.D. Supervisor with the School of Computer Science and Engineering, Northeastern University. He has authored more than 40 refereed international journals and conference papers. He has also authored two monographs published by Springer. His research work has published in high-quality international conferences, such as CIKM and DEXA, and highly-cited international journals, such as *Fuzzy Sets and Systems*, *Knowledge-Based Systems*, and *Integrated Computer-Aided Engineering*. His current research interests include RDF data management, ontology, knowledge graph, and knowledge representation and reasoning.

ZHI YANG is currently pursuing the master's degree with the School of Computer Science and Engineering, Northeastern University, China. His current research interests include knowledge graph representation learning and natural language processing.