Hindawi
Scientific Programming
Volume 2020, Article ID 4741963, 10 pages
https://doi.org/10.1155/2020/4741963

Research Article

Representation Learning of Knowledge Graphs with Embedding Subspaces

Chunhua Li,[1] Xuefeng Xian,[2] Xusheng Ai,[1] and Zhiming Cui[3]

[1]*Software and Service Outsourcing College, Suzhou Vocational Institute of Industrial Technology, Suzhou 215104, China*
[2]*School of Computer Engineering, Suzhou Vocational University, Suzhou 215104, China*
[3]*School of Electronics and Information Engineering, Suzhou University of Science and Technology, Suzhou 215009, China*

Correspondence should be addressed to Zhiming Cui; zmcui@usts.edu.cn

Received 12 February 2020; Revised 25 April 2020; Accepted 10 June 2020; Published 25 August 2020

Academic Editor: Kifayat Ullah Khan

Most of the existing knowledge graph embedding models are supervised methods and largely relying on the quality and quantity of obtainable labelled training data. The cost of obtaining high quality triples is high and the data sources are facing a serious problem of data sparsity, which may result in insufficient training of long-tail entities. However, unstructured text encoding entities and relational knowledge can be obtained anywhere in large quantities. Word vectors of entity names estimated from the unlabelled raw text using natural language model encode syntax and semantic properties of entities. Yet since these feature vectors are estimated through minimizing prediction error on unsupervised entity names, they may not be the best for knowledge graphs. We propose a two-phase approach to adapt unsupervised entity name embeddings to a knowledge graph subspace and jointly learn the adaptive matrix and knowledge representation. Experiments on Freebase show that our method can rely less on the labelled data and outperforms the baselines when the labelled data is relatively less. Especially, it is applicable to zero-shot scenario.

1. Introduction

There are various knowledge graphs constructed with great effort, such as Freebase [1] and WordNet [2], which have become the fundamental techniques for many intelligent applications such as web search and question answering [3]. However, the validity and integrity of the knowledge graphs cannot always be guaranteed. For instance, Freebase [1] currently contains entities over 80 million and thousands of relations as well as obtaining billions of facts about these entities. However, obviously, these are still only a small part of all human knowledge. In fact, the ability of Q&A system engine based on knowledge graphs is limited if it could not infer and fill in the missing facts from the obtained knowledge. Therefore, knowledge reasoning methods predicting new facts only based on the knowledge existing in knowledge graphs are desired. It is a key compensation for extracting relations from flat text.

Knowledge representation is the basis of knowledge reasoning. For example, using graph-based knowledge representation, to compute or infer a semantic relationship between entities needs to design specific graph-based algorithms. Knowledge graphs represent entities as nodes and relations as different types of edges in the form of a triple (head entity, relation, tail entity) [4]. Graph-based knowledge representation is facing many challenges, e.g., computing efficiency and data sparseness. In recent years, great progress has been made in the knowledge representation learning method based on embedding technology [5]. Embedding learning is to represent the entities and relations in knowledge graphs as low-dimensional dense real value vectors and embed the knowledge graph into a continuous vector space while keeping the structure characteristic of knowledge graphs [6]. Usually, in this low-dimensional dense real value vector space, the nearer the distance of two vectors, the higher the similarity of their semantics. Since the semantic relation of entities and relations can be calculated in the low-dimensional space in a highly efficient way and the problem of data sparseness can be resolved dramatically, the performance of knowledge extracting, fusion, and reasoning is greatly improved.

Most of the existing embedding learning methods are supervised methods and generally use the obtained structured knowledge to train models [7, 8]. The success of the supervised system largely depends on the quantity and quality of obtainable training data; it is sometimes even more important than the choice of specific learning algorithms. The cost of obtaining high quality structured knowledge is high and the knowledge graph obtained is facing a serious problem of data sparseness. Long-tail entities cannot be efficiently trained. On the other hand, unstructured text data involving relevant entities and relations information can be obtained easily in large quantities. Word vector representations of entity names can be obtained through cooccurrence mode of words in large quantities of an unlabelled text corpus. The word vectors estimated from the raw text through the natural language model are the low-dimensional dense vectors containing syntax and semantic attributes of words [9]. Therefore, since these vectors are obtained by minimizing the prediction errors in the common unsupervised tasks, they might not be the best for knowledge graphs.

This paper proposes a knowledge representation learning approach in which the unsupervised word vectors are adapted to knowledge graph subspaces with a small number of labelled data. The intuition behind our approach is the following. For a specific task, only partial latent aspects captured by the word vectors will be useful. Hence, instead of updating the word vectors directly with available labelled data, we estimate a projection of these vectors into a low-dimensional subspace. This simple method has two key advantages. One is that we get low-dimensional vectors, which are suitable for complex knowledge representation learning tasks. Another is that we can learn new vectors of all entities even if they are missing in labelled data.

2. Related Work

2.1. Structure-Based Knowledge Embedding. Structure-based embedding learning models learn the entity and relation vector representations through structure information located in triples of the knowledge graph. Most existing embedding methods belong to this category.

Most methods of this kind have been designed within the framework of relational learning from latent features by operating on latent representations of entities and relations, such as models based on collective matrix factorization [10, 11] or tensor factorization [7, 12, 13]. Many approaches have focused on increasing the expressivity and the generality of the model in energy-based frameworks for learning embeddings of entities in low-dimensional spaces [14–16]. The greater expressivity of these models comes at the expense of substantial increases in model complexity and in higher computational costs.

Compared to the early embedding models, TransE [8] is simpler and more effective. TransE regards the relations in the knowledge graph as certain translation vectors. For a triple (h, r, t), where h represents a head entity, and r is a relationship that connects h to a tail entity t. TransE use relation vector $\mathbf{r}$ as the translation between the head entity vector $\mathbf{h}$ and the tail entity vector $\mathbf{t}$. Therefore, TransE is also referred to as translation model. The basic idea of this model is to treat relation r as the translation between entities associated with r. If relation (h, r, t) is established, $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$ in the vector space, that is $\mathbf{t}$ should be the closest vector of $\mathbf{h} + \mathbf{r}$. Otherwise, $\mathbf{h} + \mathbf{r}$ should be far away from $\mathbf{t}$.

Later knowledge embedding learning models are mostly the extension based on TransE, such as TransH [17], TransR [18], TransD [19], and TransA [20]. PTransE [21] proposes a multiple-step relation path based representation learning model. TranSparse [22] deal with the heterogeneity and the imbalance of knowledge graphs with adaptive sparse matrices. The recently proposed ProjE [23] achieves the state-of-the-art performance of this branch with relatively less model parameters.

Luo et al. [24] propose a two-stage embedding scheme to improve the performance of structure-based embedding models, such as TransE, SME, and SE. It first uses a word embedding model to learn initial embeddings of entities and relations from relation paths, viewing entities and relations in the path as pseudowords. RDF2Vec [25], metapath2vec [26] and Hussein et al. [27] transform the graph data into sequences of entities and use unsupervised language model to learn entity representations considering sequences of entities as sentences. However, these method still only utilize the structure information. Recently, Dettmers et al. [28] introduce a multilayer convolutional network model, ConvE, for link prediction, which uses 2D convolution over embeddings and multiple layers of nonlinear features to model knowledge graphs.

2.2. Knowledge Embedding With Multisources Information. Structure-based knowledge embedding learning models only utilize the triple structure information of the knowledge graph and a large amount of other related information are not efficiently used yet, such as the descriptions and categories of entities and relations.

There are some studies on using the above information to learn knowledge representation. NTN [7] represents an entity as the average of its word embeddings in entity name. Wang et al. [29] align the embeddings of entities and words in the same space by utilizing entity names and Wikipedea anchors. Recently, DKRL [30] extends TransE considering text information of entity descriptions provided by knowledge bases (i.e., knowledge graph), and building entity vector representations with CNN model based on entity descriptions, which can model the word sequence information in text. When a new entity that is not in the knowledge base occurs, DKRL can generate entity vector based on its simple description. SSP model recently proposed by Xiao et al. [31] learned entity semantic vectors from entity description text by using topic model, then projected the structural loss to the semantic space codetermined by the head entity and tail entity to learn vector representations of entities and relations. The learning process of SSP model is more closely related to text information.

Most of the above models mainly use the text information of entity names and entity descriptions. This limits

the use of abundant unstructured text information in the Web.

3. Knowledge Representation Learning Based on Subspace Projection

Some work [7, 29–31] shows that learning knowledge representation through fusing multisource information can efficiently improve the performance of knowledge representations. Abundant Web text contains large quantity of unstructured knowledge related to entities. Word embedding is a kind of useful unsupervised technology and it can offer a simplified real value vector representation for each word from unlabelled free text. We use word embedding technology to obtain vector representations of entity names from abundant Web text and then adapt these vectors to a subspace which is suitable for representations of entities in the knowledge graph through projection. Therefore, knowledge representation learning can be divided into two stages: unsupervised estimation of entity name vectors and jointly supervised learning of the subspace adaptive matrix and the knowledge representation.

3.1. Estimation of Unsupervised Entity Vectors. We obtain vector representations of entity names in knowledge graphs from unlabelled Web text through unsupervised word vector learning technology and regard it as the initial vector representation estimations of entities. Word vectors are usually trained through optimizing an objective function with unlabelled data [9, 32–34]. CBOW [34] and skip-gram [9] learn word vectors capturing many syntactic and semantic relations between words. Thus, in this study we use skip-gram [9] to learn the word vector representations of entity names.

Given a sequence of training words $w_1, w_2, \ldots, w_T$, the optimizing objective of skip-gram is to maximize the average log probability:

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\le j\le c,\, j\neq 0}\log\, p\left(w_{t+j}\,\middle|\, w_t\right), \tag{1}$$

where c is the size of training context and $p(w_{t+j}\,|\,w_t)$ is defined using softmax function:

$$p\left(w_O\,\middle|\, w_I\right) = \frac{\exp\left(v'^{\,T}_{w_O} v_{w_I}\right)}{\sum_{w=1}^{W}\exp\left(v'^{\,T}_{w} v_{w_I}\right)}, \tag{2}$$

where v_w is the vector representation of word w and W is the number of words in the vocabulary.

The same as other majority of neural network model, skip-gram adopts a training method based on gradient descent. The trained model, embedding vectors $v_w \in R^{e\times 1}$, encloses information of each word w and the context around it. Therefore, these vectors can be used as input of other learning algorithms to improve performances further.

3.2. Embedding Knowledge Graphs into Subspaces. As previously mentioned, word embedding is a kind of useful unsupervised technology to obtain the initial feature vector of entity names before supervised training. These initial vector representations can be retrained with obtainable labelled data. However, the knowledge graph has serious problem of data sparseness. The quantity of entities in the database is large; however the quantity of high quality triple data related to each entity that can be obtained is relatively small with high cost. Only small quantity of supervised data causes serious overfitting. Furthermore, it is likely that only a subset of entities appears in the training triples and the vector representations of the entities missing in the training triples will never be updated. We propose a simple solution to avoid this problem.

We use $\mathbf{W^E} \in R^{e\times v}$ to denote the initial entity vector matrix obtained by skip-gram as stated in the previous section. We define the adapted embedding matrix $\mathbf{W^A}$ in subspace as the following factorization:

$$\mathbf{W^A} = \mathbf{W^S}\cdot\mathbf{W^E}, \tag{3}$$

where $\mathbf{W^S} \in R^{s\times e}$, $s \ll e$. Next, we estimate parameters of the matrix $\mathbf{W^S}$ by using the triples (labelled dataset) in the knowledge graph and keep $\mathbf{W^E}$ fixed. That is to say, we find the best mapping matrix $\mathbf{W^S}$ that projects the initial vector matrix $\mathbf{W^E}$ into the subspace with dimension s.

The idea of embedding knowledge graphs into a subspace is based on the following two key principles: (1) With reduction of embedding dimension, the model can better fit the complexity of the knowledge graph task and the amount of obtainable labelled data. (2) Through projection, all the entity vectors are indirectly updated, not only those of entities that appear in training triples.

3.3. Jointly Supervised Learning of Subspace Adaptive Matrix and Knowledge Representation

3.3.1. Jointly Learning Model. After obtaining the initial vectors of entities, we use a supervised model to jointly learn the projection matrix and knowledge representation in subspace based on the idea of subspace projection. The jointly learning model utilizes the structure information of triples existing in the knowledge graph.

The concept of embedding subspace can be applied to any structure-based knowledge representation learning models. Since currently ProjE gains the best performance in relation to reasoning task and the parameters are relatively less, we use this model together with the thought of embedding subspace as a supervised training model. This method is hereinafter referred to as sub-ProjE. Let n_e, n_r, e and s be, respectively, the number of entities, relations, unsupervised entity vector dimension, and subspace dimension. The number of parameters of ProjE is $n_e \times s + n_r \times s + 5s$. The number of parameters of sub-ProjE is $s \times e + n_r \times s + 5s$. Since $e \ll n_e$, the parameter size of sub-ProjE is much smaller than that of ProjE.

We consider the relation reasoning as an entity ranking problem, which takes a partial triple as input and produces a ranked list of candidate entities as output.

Definition 1. (entity ranking problem). Given a knowledge graph $\mathcal{G} = \{\mathbf{E}, \mathbf{R}\}$ and an input tripe $(h, r, ?)$, the entity ranking problem attempts to find the optimal ordered list such that $\forall e_j \forall e_i ((e_j \in E\_ \wedge e_i \in E_+) \longrightarrow e_i \prec e_j)$, where $\mathbf{E}_+ = \{e \in \{e_1, e_2, \ldots, e_l\} \mid (h, r, e) \in \mathcal{G}\}$, $\mathbf{E}_- = \{e \in \{e_{l+1}, e_{l+2}, \ldots, e_{|E|}\} \mid (h, r, e) \notin \mathcal{G}\}$.

Similarly, we can easily substitute $(h, r, ?)$ for $(?, r, t)$. The key thought of ProjE is as follows: given two input vectors, regard prediction task as a ranking problem, keep the target of optimizing as the overall order of candidate entities, in which the entities in the front are correct entities. To generate this ordered list, we project each candidate vector to the objective vector defined by a combination operator and two input vectors. The combination operator is defined as follows:

$$\mathbf{e} \oplus \mathbf{r} = \mathbf{D}_e e + \mathbf{D}_r r + \mathbf{b}_c, \tag{4}$$

where $\mathbf{e}$ and $\mathbf{r}$ are the representations of entities and relations in embedding space, s is the dimension of the embedding space, $\mathbf{D}_e$ and $\mathbf{D}_r$ is the diagonal matrix of $s \times s$, which serve as global entity and relation weights, respectively, $\mathbf{b}_c \in R^s$ is the combination bias.

With this combination operator, we can define vector project function as follows:

$$h(\mathbf{e}, \mathbf{r}) = g\left(\mathbf{W}^c f(\mathbf{e} \oplus \mathbf{r}) + b_p\right), \tag{5}$$

where f and g are activation functions, $\mathbf{W}^c \in \mathrm{R}^{c \times s}$ is the candidate entity matrix, b_p is the projection bias, and c is the quantity of candidate entities. $h(\mathbf{e}, \mathbf{r})$ represents the ranking score vector, in which each element represents the similarity between candidate entity in $\mathbf{W}^c$ and the combined input vector.

$\mathbf{W}^c$ is the candidate entity matrix which contains c rows that exist in the entity vector matrix $\mathbf{W}^{\mathbf{E}}$ (i.e., $\mathbf{W}^{\mathbf{S}} \cdot \mathbf{W}^{\mathbf{E}}$ in knowledge graph subspace). So, $\mathbf{W}^c$ does not introduce any new variables into the model. The model can be regarded as a neural network that contains an entity vector projection layer, a combination layer, and an output projection layer. Figure 1 explains the architecture of this model through an example with input (?, City Of, Illinois). Given a tail entity Illinois and a relation City Of, our task is to calculate the scores of each head entity. In order to make it clear, we only demonstrate two candidate entities in Figure 1. However, in fact $\mathbf{W}^c$ may contain candidate entities of any quantity.

Compared to a conventional knowledge embedding model, this model has two main differences. First, the input layer is factorized into two components, the initial vector representations attained in unsupervised stage, $\mathbf{W}^{\mathrm{E}}$, and the projection matrix $\mathbf{W}^{\mathbf{S}}$. Second, the size of the subspace, in which the initial vectors are projected, is much smaller than that of the initial embedding space with typical reductions above one order of magnitude. Same as the usual neural network model, all the parameters can be trained with gradient descent methods through backpropagation.

3.3.2. Ranking Method and Loss Function. Following ProjE, we construct a binary label vector, in which all entities in $\mathbf{E}\_$ have a value of 0 and all entities in $\mathbf{E}_+$ have a value of 1, then maximize the likelihood between ranking score vector $h(\mathbf{e}, \mathbf{r})$ and the binary label vector. The loss function is defined as follows:

$$L(\mathbf{e}, \mathbf{r}, \mathbf{y}) = -\sum_i^{|y|} \frac{1(y_i = 1)}{\sum_i 1(y_i = 1)} \log\left(h(\mathbf{e}, \mathbf{r})_i\right), \tag{6}$$

where $\mathbf{e}$ and $\mathbf{r}$ are the vector representation of input training sample, $y \in R^c$ is the binary label vector, where $y_i = 1$ means candidate entity i is positive, the objective probability of a positive candidate (objective value) is 1 divided by the total number of positive candidates. We regard softmax and tanh function as $g(\cdot)$ and $f(\cdot)$, respectively, then the ranking score of the ith candidate entity is as follows:

$$h(\mathbf{e}, \mathbf{r})_i = \frac{\exp\left(\mathbf{W}^c_{[i,:]} \tanh(e \oplus r) + b_p\right)}{\sum_j \exp\left(\mathbf{W}^c_{[i,:]} \tanh(e \oplus r) + b_p\right)}, \tag{7}$$

where $\mathbf{W}^c_{[i,:]}$ represents the ith candidate in the candidate entity matrix.

3.3.3. Algorithms. Since the quantity of candidate entities (i.e., rows of $\mathbf{W}^c$) is large, we use candidate sampling to reduce the number of candidate entities in the training phase. Given an entity $\mathbf{e}$, a relation $\mathbf{r}$ and a binary label vector $\mathbf{y}$, we calculate projections of all positive candidates. For negative candidates, we only calculate projections of a sampled subset. We take negative candidate samples based on the binomial probability distribution $B(1 - p_y)$, in which p_y is the probability of a negative sample which might be sampled, $1 - p_y$ is the probability of a negative sample that is sampled. For each negative candidate in $\mathbf{y}$, we sample a value from $B(1 - p_y)$ to determine whether this candidate is included in the candidate entity matrix $\mathbf{W}^c$ or not.

The complete training process is demonstrated in Algorithm 1. Given training triples $\mathbf{T}$, we first choose at random to replace head entity or tail entity to construct real training dataset and then generate positive and negative samples from $\mathbf{T}$ according to sampling strategy. Next, calculate loss and update parameters for each minibatch in the newly generated training dataset. Among this, $\circ$ is Hadamard product and $\times$ is matrix product.

4. Experiments

We evaluate our model with entity prediction tasks and compare the performance against the native ProjE using experimental procedures, datasets, and metrics established in the related work. We also give the results of TransE [8] and TransH [17] implemented by [18] on our datasets.

4.1. Experimental Setting

4.1.1. Dataset. For evaluating our proposed model, we use FB15K and FB15K-237 datasets to conduct experiments.

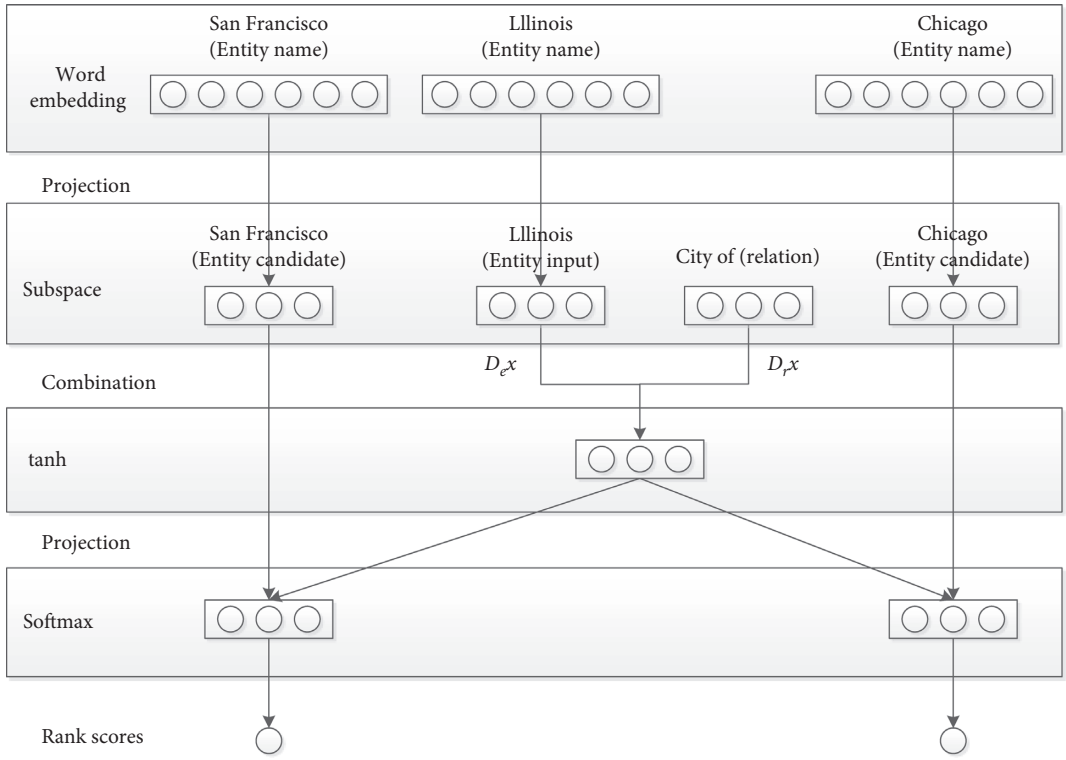


Figure 1: Illustration of the sub-ProjE model on entity prediction task.

The FB15K dataset released by [8] is a subset of Freebase [1], which contains 592,213 triples and involves 14,951 entities and 1345 relations. The initial entity vector is indicated with the entity name vector unsupervised pretrained by word2vec [9] with a large-scale web corpus. The dimension of the initial entity vector is 1000. In order to ensure each entity has a pretrained initial vector representation, we deleted 1423 entities without entity name vectors from FB15K and removed triples about these entities accordingly. The finally retained training set includes 364,424 triples, validation set includes 37,905 triples and the test set includes 44,565 triples, which totally involves 13,528 entities and 1345 relations.

FB15K-237 introduced by Toutanova and Chen [35] is a subset of FB15K where inverse relations are removed. We remove entities without entity name vectors too, the same as FB15K. The finally resulted FB15K-237 dataset contains 211,380 training triples, 21,981 validation triples, and 25,666 test triples, which totally involves 13,528 entities and 237 relations.

For zero-shot scenario, we divide entities into two groups: training entities (10,000) and test entities (3,528), while ensuring the training set and validation set only contains triples whose head entity and tail entity are both in training entity group, and the triple in test set has at least one entity in a test entity group. Finally, the FB15K dataset for zero-shot scenario has 201,272 training triples, 20,968 validation triples, and 3,012 test triples, respectively. The FB15K-237 dataset for zero-shot scenario has 117,785 training triples, 12,196 validation triples, and 1,762 test triples, respectively. Table 1 shows the statistical properties of datasets.

4.1.2. Parameter Setting. In the supervised training phase, we apply the default setting the same as ProjE: using Adam [36] as the stochastic optimizer with hyperparameter settings of $\beta_1 = 0.9$, $\beta_1 = 0.999$, $\varepsilon = 1e^{-8}$; L_1 regularized to all parameters during the training and dropout layer on top of the combination operator to avoid overfitting. Other parameters are set as follows: learning rate $l_r = 0.01$, batch size $b = 200$, regularized parameter $\alpha = 1e^{-5}$, dropout probability $p_d = 0.5$, negative sample sampling probability $p_y = 0.5$.

4.2. Experimental Results

4.2.1. Evaluation Protocol. In the entity prediction tasks, we predict the missing head entity or tail entity in the triples by ranking all of the entities in the knowledge graph. Given a test triple (h, r, t), we remove the head or tail entity and then replace it with each entity in the knowledge graph and calculate the ranking score, then rank these replacing entities in descending order and record the ranking of the right entities. Following [8], we use mean rank, HITS@k, filtered mean rank, and filtered HITS@k as our evaluation metrics.

4.2.2. Results. Table 2 shows the results of different models in entity prediction tasks trained on FB15K with different training set sizes. We can see from Table 2 that the Mean Rank of sub-ProjE is dramatically superior to that of ProjE, TansH, and TransE when the training data becomes less. Both sub_ProjE and ProjE outperform TransE and TransH.

Input: Training triples $T = \{(h, r, t)\}$, entities $\mathbf{E}$, initial entity matrix $\mathbf{W}^{\mathbf{E}}$, relations R, embedding subspace dimension s, dropout probability p_d, candidate sampling rate p_y, regularize parameter α
Output: subspace adaptive matrix $\mathbf{W}^{\mathbf{s}}$, relation embedding matrix WR, combination operators $\mathbf{D}_{eh}$, $\mathbf{D}_{rh}$, $\mathbf{D}_{et}$, $\mathbf{D}_{rt}$
Algorithm sub-ProjE ($\mathbf{T}$, $\mathbf{E}$, $\mathbf{W}^{\mathbf{E}}$, $\mathbf{R}$, s, p_d, p_y, α)
(1) Initializing adaptive matrix $\mathbf{W}^{\mathbf{s}}$, relation matrix $\mathbf{W}^{\mathbf{R}}$, combination operators (diagonal matrices) $\mathbf{D}_{eh}$, $\mathbf{D}_{rh}$, $\mathbf{D}_{et}$, $\mathbf{D}_{rt}$ with uniform distribution $(-6/\sqrt{s}, 6/\sqrt{s})$.
(2) **Loop**/*A training iteration/epoch*/
(3) $\mathbf{T}^h \leftarrow$, $\mathbf{C}^h \leftarrow$, $\mathbf{T}^t \leftarrow$, $\mathbf{C}^t \leftarrow$;//training data
(4) **for** $(h, r, t) \in \mathbf{T}$ do/*construct training data using all train triples*/
(5) $e \leftarrow \text{random}(h, t)$
(6) **if** $e == h$ **then**/*tail is missing*/
(7) $\mathbf{T}^h.\text{add}([e, r])$
(8) $\mathbf{C}^h.\text{add}(\{t' \mid (h, r, t') \in \mathbf{T}\} \cup \text{sample}(\mathbf{E}, \mathbf{p}_\mathbf{y}))$/*all positive tails from T and some sampled negative candidates*/
(9) **else**/*head is missing*/
(10) $\mathbf{T}^t.\text{add}([e, r])$
(11) $\mathbf{C}^t.\text{add}(\{h' \mid (h', r, t) \in \mathbf{T}\} \cup \text{sample}(\mathbf{E}, \mathbf{p}_\mathbf{y}))$/*all positive heads from T and some sampled negative candidates*/
(12) **end if**
(13) **end for**
(14) **for each** $(\mathbf{T}_b^h, \mathbf{C}_b^h, \mathbf{T}_b^t, \mathbf{C}_b^t) \subset (\mathbf{T}^h, \mathbf{C}^h, \mathbf{T}^t, \mathbf{C}^t)$ **do**/*minibatches*/
(15) $l \leftarrow 0$
(16) **for each** $(\mathbf{t}^h, \mathbf{c}^h, \mathbf{t}^t, \mathbf{c}^t) \in (\mathbf{T}_b^h, \mathbf{C}_b^h, \mathbf{T}_b^t, \mathbf{C}_b^t)$ do/*training instance*/
(17) $O_h \leftarrow \text{softmax}((\mathbf{W}^s \times \mathbf{W}^{\mathbf{E}})_{[\mathbf{c}^t,:]} \times \tanh(\text{dropout}(p_d, \mathbf{D}_{et} \times ((\mathbf{W}^s \times \mathbf{W}^{\mathbf{E}})_{[\mathbf{t}^t[0],:]})^T + \mathbf{D}_{rt} \times (\mathbf{W}^{\mathbf{R}}_{[\mathbf{t}^t[1],:]})^T + b_c)) + b_p)$
(18) $O_t \leftarrow \text{softmax}((\mathbf{W}^s \times \mathbf{W}^{\mathbf{E}})_{[\mathbf{c}^h,:]} \times \tanh(\text{dropout}(p_d, \mathbf{D}_{eh} \times ((\mathbf{W}^s \times \mathbf{W}^{\mathbf{E}})_{[\mathbf{t}^h[0],:]})^T + \mathbf{D}_{rh} \times (\mathbf{W}^{\mathbf{R}}_{[\mathbf{t}^h[1],:]})^T + b_c)) + b_p)$
(19) $l = l - \sum(\{1((h, \mathbf{t}^t[1], \mathbf{t}^t[0]) \in \mathbf{T}) \mid h \in \mathbf{c}^t\} \circ \log(\mathrm{O_h})) - \sum(\{1((\mathbf{t}^h[0], \mathbf{t}^h[1], t) \in \mathbf{T}) \mid t \in \mathbf{c}^h\} \circ \log(\mathrm{O_h}))$
(20) **end for**
(21) $l_r \leftarrow \text{Regu}_1(\mathbf{W}^s) + \text{Regu}_1(\mathbf{W}^{\mathbf{R}}) + \text{Regu}_1(\mathbf{D}_{eh}) + \text{Regu}_1(\mathbf{D}_{rh}) + \text{Regu}_1(\mathbf{D}_{et}) + \text{Regu}_1(\mathbf{D}_{rt})$
(22) Update all parameters w.r.t $l + \alpha l_r$
(23) **end for**
(24) **EndLoop**

Algorithm 1: Embedding knowledge graphs into subspace.

Table 1: Statistical properties of the dataset.

Dataset	Relation	Entity	Training set	Validation set	Test set
FB15K	1345	13528	364424	37905	44565
FB15K-zero-shot	1345	10000 (train)/3528 (test)	201272	20968	3012
FB15K-237	237	13528	211380	21981	25666
FB15K-237-zero-shot	237	10000 (train)/3528 (test)	117785	12196	1762

Since the training data is less, the filtered Mean Rank is similar to the original Mean Rank. When the training set is very large, the performances of ProjE, TransE, and TransH are superior to that of sub-ProjE, which verifies the fact that sub-ProjE is applicable to less training data scenario. HITs@10 of ProjE is superior to sub-ProjE. This is because ProjE only partially updates the entity vectors during training. Therefore, partially the accuracy of ProjE is higher than that of sub-ProjE, but on the whole, the performance of sub-ProjE is superior to that of ProjE. Table 3 shows the results of different models in entity prediction tasks trained on FB15K-237 with different training set sizes. The results are similar to the results of FB15K.

In the zero-shot scenario, at least one entity of a tested triple is not in the knowledge graph (i.e., the training set). The original ProjE method cannot deal with this situation since it cannot generate entity representation that is not in the knowledge graph. TransE and TransH cannot apply to zero-shot scenario too for the same reason. Our sub-ProjE method is capable of dealing with this situation because it indirectly updates the missing entity representation during training. Table 4 shows the results of zero-shot scenario experiments on FB15K and FB15K237. We can see from Table 4 that sub-ProjE can deal with the entity prediction in zero-shot scenario even when the training data is very limited.

Table 2: Prediction results with different training set sizes on FB15K.

		Mean rank		HITs@10 (%)	
Training set size	Method	Raw	Filtered	Raw	Filtered
Train = 5000	TransE	5042	5030	10.4	10.7
	TransH	5125	5113	10.2	10.5
	ProjE	1340	1328	24.3	32.8
	Sub-ProjE	534	522	19.2	19.9
Train = 10000	TransE	3372	3358	16.0	16.6
	TransH	3388	3375	16.2	16.8
	ProjE	717	705	32.9	42.8
	Sub-ProjE	408	395	22.8	23.7
Train = all	TransE	165	98	53.7	70.9
	TransH	166	100	58.0	70.3
	ProjE	157	98	42.5	66.1
	Sub-ProjE	326	266	24.8	34.3

Table 3: Prediction results with different training set sizes on FB15K-237.

		Mean rank		HITs@10 (%)	
Training set size	Method	Raw	Filtered	Raw	Filtered
Train = 5000	TransE	4559	4544	13.6	14.0
	TransH	4568	4554	13.4	13.9
	ProjE	1155	1141	29.6	41.6
	Sub-ProjE	447	434	19.5	20.3
Train = 10000	TransE	2902	2886	18.9	19.9
	TransH	2972	2955	18.5	19.5
	ProjE	614	599	38.2	52.2
	Sub-ProjE	377	362	23.7	24.9
Train = all	TransE	297	219	42.9	58.3
	TransH	293	217	42.5	58.0
	ProjE	164	98	40.8	64.9
	Sub-ProjE	313	246	25.7	35.3

Table 4: Zero-shot scenario results of sub-ProjE with different training set sizes.

		Mean rank		HITs@10 (%)	
Training set size	Dataset	Raw	Filtered	Raw	Filtered
Train = 5000	FB15K-237	767	769	12.2	12.2
	FB15K	748	746	12.4	12.5
Train = 10000	FB15K-237	699	698	13.0	13.1
	FB15K	669	667	14.4	14.4
Train = all	FB15K-237	524	523	16.3	16.4
	FB15K	531	529	17.4	17.5

With the increase of training data, the performance of the method is further improved.

In order to further analyze the stability of sub-ProjE model, we give the mean rank and HITs@10 results of the first 30 iterations on FB15K in a common scenario and zero-shot scenario, as Figures 2 and 3 show. We can see from Figure 2 that the larger the training dataset, the faster the sub-ProjE model converges. The performance of sub-ProjE model becomes stable after the first several iterations when all the training dataset is put into training. When the data is less, sub-ProjE converges slower and becomes stable after over ten times of iteration. The convergent speed of the zero-shot scenario is the same as the general scenario. We get similar results on FB15K-237. This shows sub-ProjE model is applicable to the zero-shot scenario.

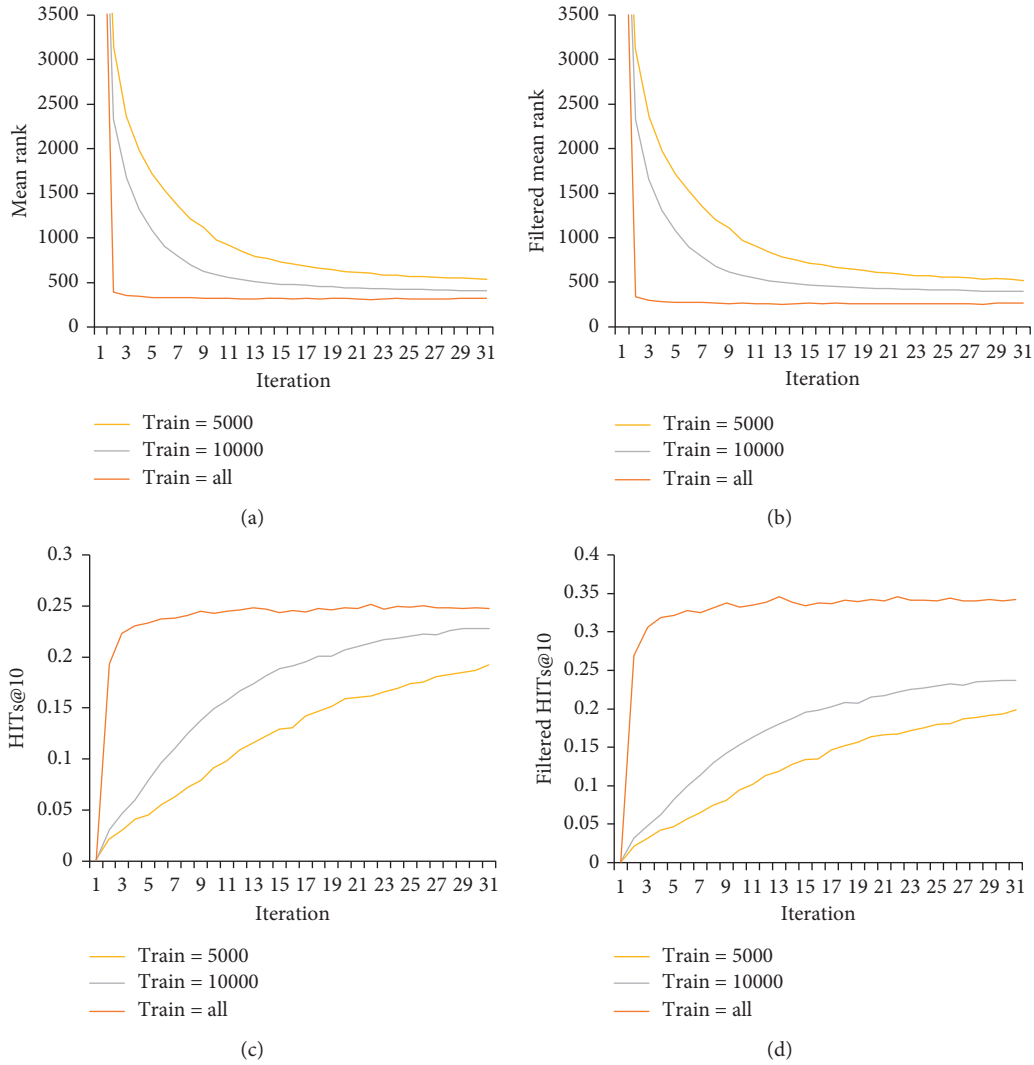
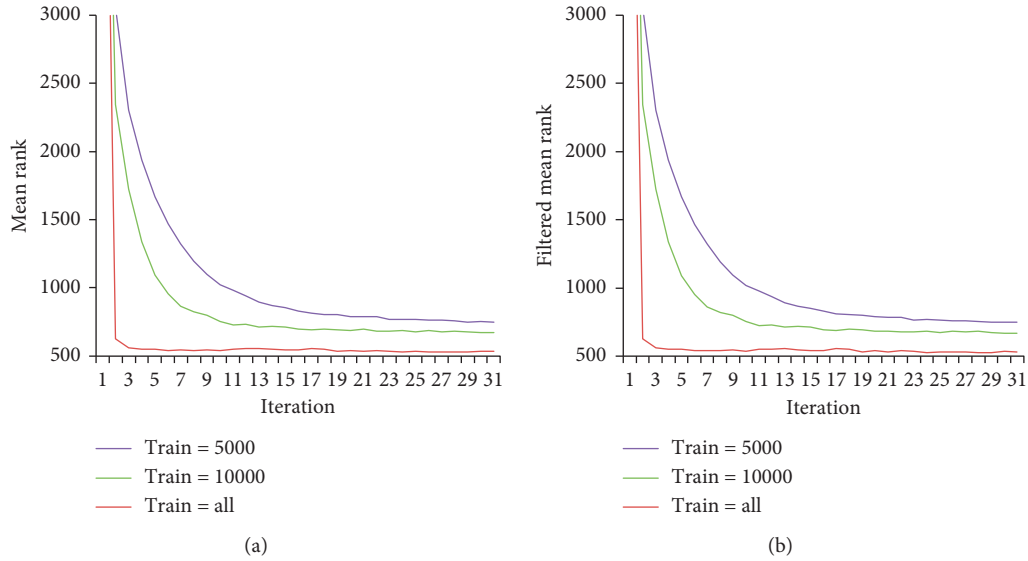


Figure 2: Training process of sub-ProjE model in general scenario. (a) Mean Rank. (b) Filtered Mean Rank. (c) HITs@10. (d) Filtered HITs@10.



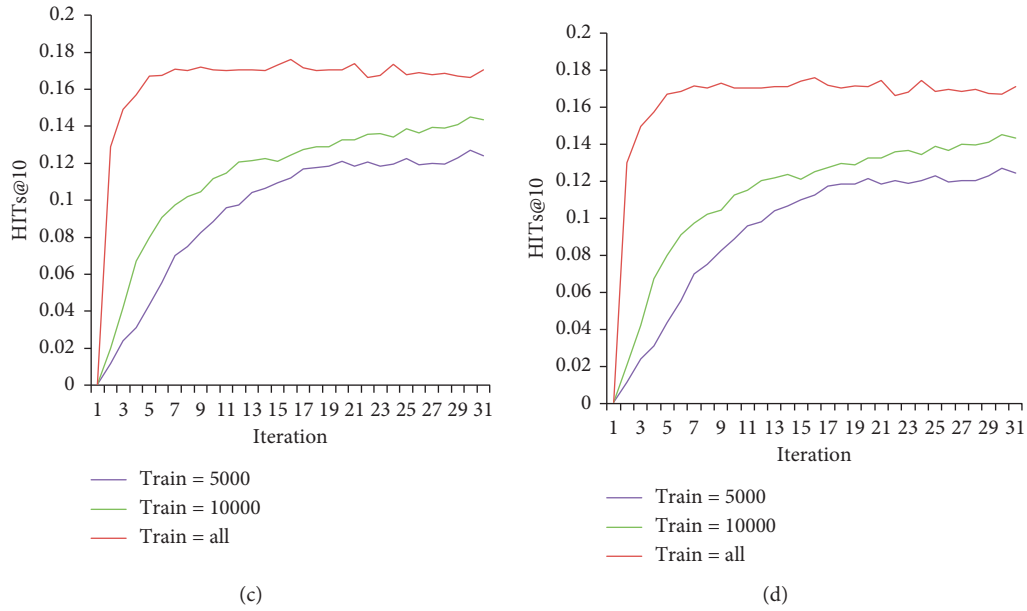


Figure 3: Training process of sub-ProjE model in zero-shot scenario. (a) Mean Rank. (b) Filtered Mean Rank. (c) HITs@10. (d) Filtered HITs@10.

5. Conclusions

This paper proposes a new knowledge representation learning method utilizing unsupervised entity name vectors. The basic idea is to seek the subspace projection of unsupervised entity vectors in knowledge representation tasks. This method allows indirect update of entity vectors that do not appear during the process of training and applicable to the case that only a few labelled data can be obtained. Experiments on Freebase verify the effectiveness of this method. Results show that the performance of this simple method surpasses the best existing knowledge representation learning model in case the training data is less, and furthermore, it can be applied to zero-shot scenarios.

Data Availability

The datasets used in this paper to produce the experimental results are publicly available. FB15k and FB15k-237 can be downloaded from http://openke.thunlp.org. The unsupervised pretrained entity vectors can be downloaded from http://code.google.com/p/word2vec.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant no. 61876217, Suzhou Science and Technology Plan Project under Grant no. SYG201903, Fundamental Computing Education Research Project of Association of Fundamental Computing Education in Chinese Universities under Grant nos. 2018-AFCEC-328 and 2019-AFCEC-288, and Research Funds of Suzhou Vocational Institute of Industrial Technology under Grant no. 2019kyqd001.

References

[1] K. Bollacker, C. Evans, P. Paritosh et al., "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pp. 1247–1250, Vancouver, Canada, June 2008.

[2] G. A. Miller, "WordNet: a lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[3] C. Unger, L. Bühmann, J. Lehmann et al., "Template-based question answering over RDF data," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 639–648, Lyon, France, April 2012.

[4] M. Nickel, K. Murphy, V. Tresp et al., "A review of relational machine learning for knowledge graphs," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2015.

[5] 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.

[6] A. García-Durán, A. Bordes, N. Usunier, and Y. Grandvalet, "Combining two and three-way embedding models for link prediction in knowledge bases," *Journal of Artificial Intelligence Research*, vol. 55, pp. 715–742, 2016.

[7] R. Socher, D. Chen, C. D. Manning et al., "Reasoning with neural tensor networks for knowledge base completion," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 926–934, Lake Tahoe, NV, USA, December 2013.

[8] A. Bordes, N. Usunier, A. Garcia-Duran et al., "Translating embeddings for modeling multi-relational data," in *Proceedings of the Advances in Neural Information Processing*

Systems, pp. 2787–2795, Lake Tahoe, NV, USA, December 2013.

[9] T. Mikolov, I. Sutskever, K. Chen et al., "Distributed representations of words and phrases and their compositionality," in *Proceedings of the Advances in Neural Information Processing Sytems*, pp. 3111–3119, Lake Tahoe, NV, USA, December 2013.

[10] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 650–658, Las Vegas, NV, USA, August 2008.

[11] M. Nickel, V. Tresp, and H. P. Kriegel, "A three-way model for collective learning on multi-relational data," *ICML*, vol. 11, pp. 809–816, 2011.

[12] M. Nickel, V. Tresp, and H. P. Kriegel, "Factorizing yago: scalable machine learning for linked data," in *Proceedings of the 21st International Conference on World Wide Web*, pp. 271–280, Lyon, France, April 2012.

[13] R. Jenatton, N. L. Roux, A. Bordes et al., "A latent factor model for highly multi-relational data," in *Proceedings of the Advances in Neural Information Processing Systems*, pp. 3167–3175, San Francisco, CA, USA, Decemeber 2012.

[14] A. Bordes, J. Weston, R. Collobert et al., "Learning structured embeddings of knowledge bases," in *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, San Francisco, CA, USA, August 2011.

[15] A. Bordes, X. Glorot, J. Weston, and Y. Bengio, "A semantic matching energy function for learning with multi-relational data," *Machine Learning*, vol. 94, no. 2, pp. 233–259, 2014.

[16] D. Chen, R. Socher, C. D. Manning et al., "Learning new facts from knowledge bases with neural tensor networks and semantic word vectors," 2013, http://arxiv.org/abs/1301.3618.

[17] Z. Wang, J. Zhang, J. Feng et al., "Knowledge graph embedding by translating on hyperplanes," in *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, Québec, Canada, July 2014.

[18] Y. Lin, Z. Liu, M. Sun et al., "Learning entity and relation embeddings for knowledge graph completion," in *Proceedings of the AAAI*, pp. 2181–2187, Austin, TX, USA, January 2015.

[19] G. Ji, S. He, L. Xu et al., "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*, vol. 1, pp. 687–696, Beijing, China, July 2015.

[20] Y. Jia, Y. Wang, H. Lin et al., "Locally adaptive translation for knowledge graph embedding," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA, February 2016.

[21] Y. Lin, Z. Liu, H. Luan et al., "Modeling relation paths for representation learning of knowledge bases," 2015, http://arxiv.org/abs/1506.00379.

[22] G. Ji, K. Liu, S. He et al., "Knowledge graph completion with adaptive sparse transfer matrix," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, Phoenix, AZ, USA, February 2016.

[23] B. Shi and T. Weninger, "ProjE: embedding projection for knowledge graph completion," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, San Francisco, CA, USA, February 2017.

[24] Y. Luo, Q. Wang, B. Wang et al., "Context-dependent knowledge graph embedding," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1656–1661, Lisbon, Portugal, September 2015.

[25] P. Ristoski and H. Paulheim, "RDF2Vec: RDF graph embeddings for data mining," *Lecture Notes in Computer Science*, Springer, Berlin, Germany, pp. 498–514, 2016.

[26] Y. Dong, N. V. Chawla, A. Swami et al., "Metapath2vec: Scalable Representation Learning for Heterogeneous Networks," in *Proceedings of the Knowledge Discovery and Data Mining*, pp. 135–144, Jeju, South Korea, May 2017.

[27] R. Hussein, D. Yang, P. Cudremauroux et al., "Are meta-paths necessary?: revisiting heterogeneous graph embeddings," in *Proceedings of the Conference on Information and Knowledge Management*, pp. 437–446, Torino, Italy, October 2018.

[28] T. Dettmers, P. Minervini, P. Stenetorp et al., "Convolutional 2d knowledge graph embeddings," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, LA, USA, February 2018.

[29] Z. Wang, J. Zhang, J. Feng et al., "Knowledge graph and text jointly embedding," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1591–1601, Doha, Qatar, October 2014.

[30] R. Xie, Z. Liu, J. Jia et al., "Representation learning of knowledge graphs with entity descriptions," in *Proceedings of the AAAI*, pp. 2659–2665, Phoenix, AZ, USA, February 2016.

[31] H. Xiao, M. Huang, and X. Zhu, "SSP: semantic space projection for knowledge graph embedding with text descriptions," 2016, http://arxiv.org/abs/1604.04835.

[32] Y. Bengio, R. Ducharme, P. Vincent et al., "A neural probabilistic language model," *Journal of Machine Learning Research*, vol. 3, pp. 1137–1155, 2003.

[33] R. Collobert, J. Weston, L. Bottou et al., "Natural language processing (almost) from scratch," *Journal of Machine Learning Research*, vol. 12, pp. 2493–2537, 2011.

[34] T. Mikolov, K. Chen, G. S. Corrado et al., "Efficient estimation of word representations in vector space," in *Proceedings of the International conference on learning representations*, Scottsdale, AZ, USA, May 2013.

[35] 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, July 2015.

[36] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proceedings of the International conference on learning representations*, San Diego, CA, USA, May 2015.