The Distributed Representation of Knowledge Graphs Based on Pseudo-Siamese Network

Zhuo Wei*, Ye Zhang*, Fan Wangb and Shuai Liuc
Changchun Institute of Technology, Jilin, China

*Corresponding author e-mail: weizhuo@foxmail.com, *81363846@qq.com,
b505009596@qq.com, c3996881@qq.com

Abstract. This paper proposes to transform the (head entity, relation) and tail entity into the same feature space through the Pseudo-Siamese network, and calculate the similarity between the two parts in this feature space, embedding vector of entity and relation have been optimization for constraint conditions. In this paper, the triple is regarded as the abstraction of the answer pair in the factual simple question-answering. According to the corresponding relation of the answer, the corresponding relation between the (head entity, relation) and the tail entity in the triple is obtained, and the constraint characteristics of the elements in the triple are modeled. And then by constructing inverse relations to build a new triple, thus the number of training samples is expanded to improve the learning results of the model.

1. Introduction
In this paper, the distributed representation of the knowledge graph is carried out by learning the constraint characteristics of the internal elements of the triple, in order to preserve the local structural characteristics of the knowledge graph in the embedded vector space. According to the corresponding relationship between triples and factual simple question and answer in knowledge graph, triples can be regarded as an abstraction of question answer pairs, so the combination of head entities and relationships in triples often contains the same semantics as tail entities. Therefore, in this paper, the ternary is divided into (head entity, relation) and tail entity, and the two parts are transformed into the same feature space by pseudo-twin network to calculate their similarity, and the training objective function based on multi-classification problem is obtained. Automatic question answering based on knowledge graph takes knowledge graph as the answer base of questions. In particular, for factual simple questions and answers, once linked to the relevant triple, the answer to the question is available within the triplet.

2. Relevant Research
This paper us deal with (head entity, relation) and tail entity through the Siamese network. Siamese network used to measure the similarity between two inputs, which is widely used in computer vision, such as image matching [1], Video tracking [2] and face recognition [3], and so on. The structure of Pseudo-Siamese network is shown in Figure 1. The network contains two branches with the same structure. When the weights of two branches are shared, they are strictly Pseudo-Siamese network, which are used to process two "relatively similar" inputs, such as a person's different photos, two semantically similar words, etc. In addition, there are "some differences” in input, such as questions in
sentence form and answers in word form, pictures and corresponding titles. For this kind of input, pseudo-Siamese network can be used. The difference between pseudo-Siamese network and strictly pseudo-Siamese network is that the weights of the two branches are not shared. The purpose of this paper is to compare the similarity between the head entity and the tail entity. Although they are similar in semantics, they are different in composition, so it is suitable to use pseudo twin network.

ProjE [4] and NAM [5] learn the weighted fusion result of head entity and relationship adaptively through neural network. The similarity between the fusion result and tail entity constitutes the basis of training objective function of the model. The difference between our work and the above method is that we regard (head entity, relation) and tail entity as the input of two branches of Pseudo-Siamese network, first transform them into the same feature space, then compare the similarity, so as to eliminate the spatial heterogeneity between the fusion result of head entity and relation and the embedding vector of tail entity. In addition, compared with the translation based model, this paper enhances the learning ability of the model through the nonlinear transformation and adaptive learning of the Pseudo-Siamese network.

![Network structure of Siamese network and pseudo-Siamese network](image)

**Figure 1.** Network structure of Siamese network and pseudo-Siamese network

3. Model Architecture

3.1. The structure of Pseudo-Siamese network

Given (head entity, relation) and candidate tail entity, Dis-Siamese transforms these two parts into the same feature space through two branches of Pseudo-Siamese network. By calculating the similarity between them, the confidence score of candidate triples is obtained. The candidate entities are sorted accordingly. The aim of Dis-Siamese is to make the correct candidate entities ahead of other entities by optimizing entity and relation embedding vectors and model weight parameters.

The network structure of Dis-Siamese is shown in Figure 2 including input layer, Pseudo-Siamese branches and decision module. Given head entity h and relation r, place the entities in the candidate entity set in the tail entity position, constitute candidate triples (h, r, c_i), c_i represents the Number i candidate entity. In the input layer, the candidate triples are divided into (h, r, c) and c, and each element is mapped to the corresponding embedding vector (h, r) and c. Embedding vector dimension is d. In addition, as shown previously, in this paper, a new triple is constructed by inverse relation, that is, the same principle is mapped to sum c in the input layer.
Since the calculation process of these two types of input is similar, we take the input candidate triples \((h, r, c_i)\) as an example to explain later. \((h, r,)\) and \(c_i\) enter into two branches of pseudo twin network respectively, and transform feature space through nonlinear layer. The output of branch 1 is calculated as follows:

\[
\mathbf{z}_1 = \mathbf{f} \left( \mathbf{W}_1 \mathbf{h} + \mathbf{b}_1 \right) \in \mathbb{R}^{d \times 2d}
\]

\(\mathbf{W}_1\) is the weight matrix, \(\mathbf{b}_1\) is the Offset item. \([x_1; x_2]\) means to splice two vectors. \(\mathbf{f} (\cdot)\) represents tanh nonlinear activation function. Branch 2 has the same structure as branch 1. The difference is that the parameters of weight matrix and offset term are different. Specifically, the output of branch 2 is calculated according to the following formula:

\[
\mathbf{z}_2 = \mathbf{f} \left( \mathbf{W}_2 \mathbf{c}_i^T + \mathbf{b}_2 \right) \in \mathbb{R}^{d \times 2d}
\]

\(\mathbf{W}_2\) is the weight matrix, \(\mathbf{b}_2\) is the Offset item. The decision module calculates the confidence score of the input triplet through the output similarity of the Pseudo-Siamese network. After the output \(\mathbf{Z}_1\) and \(\mathbf{Z}_2\) of the two branches of the Pseudo-Siamese network are normalized, the similarity score, the confidence score of the input triple, is obtained as follows:

\[
\mathbf{s}(h, r, c) = g(\mathbf{z}_1 \mathbf{z}_2^T) \quad g(\cdot) \text{ represents nonlinear activation function.}
\]

Select according to the following training objective function.

3.2. Sorting Method and Training Objective Function

The distributed representation of knowledge graph is often used to complete the knowledge graph, such as link prediction. Its essence is to sort the candidate entities or relationships, so that the correct candidate entities or relationships are in the front as much as possible. Among them, the definition of entity ordering problem is as follows:

Definition 1: Given the knowledge graph, \(\mathcal{G} = \mathcal{E}, \mathcal{R}\) and input triple entity scheduling problem aims to optimize the candidate entity scheduling. \(\forall e_1 \forall e_j ((e_i \in \mathcal{E}_+ \cup e_j \in \mathcal{E}_-) \rightarrow e_i \sim e_j)\), \(\mathcal{E}_+ = \{e \in \mathcal{E} | (h, r, e) \in \mathcal{G}\}\) and \(\mathcal{E}_- = \{e \in \mathcal{E} | (h, r, e) \notin \mathcal{G}\}\).

When the input triple is \((?, r, t)\), the problem of entity sorting can be transformed into the problem of \((?, \hat{r}, t)\) entity sorting through inverse relation. The relational ordering is designed to find the optimal relational ordering for the input triple. In this paper, the training objective function of the model is established based on the above-mentioned entity ranking problem, and the correct candidate entity is
ranked ahead of other candidate entities by optimizing the entity and relation embedding vector and model parameters. There are usually two solutions to this sorting problem: pairwise ranking and listwise ranking. Correspondingly, in this paper, we use Dis-Siamese_pointwise and Dis-Siamese_listwise.

3.3. Candidate Sampling Method

The denominator in formula will involve the scores of all candidate entities, which brings about great computational complexity. If reduced the number of Candidate entities in the training process, the computational complexity and training time of the model can be reduced. Therefore, this paper reduces the number of candidate entities by sampling the candidate set, and specifically uses the negative sampling method in word2vec for reference.

Given the head entity h, the relation r and the binary marker vector y, the input candidate set consists of two parts: all the correct candidate entities and the negative sample subset obtained according to the negative sampling probability distribution. Specifically, P is (0, 1) the Bernoulli distribution B(1, P_y), and p represents the probability of negative samples being sampled and 1-P_y represents the probability of negative samples not being sampled. For each negative sample corresponding to the element whose value is 0 in the binary marker vector y, the distribution is used to determine whether to include it in the input candidate entity set.

4. Experimental analysis

In this paper, we use Python to implement Dis-Siamese algorithm on Tensorflow platform, and evaluate the performance of the algorithm through two knowledge graph completion tasks: entity prediction and relationship prediction. The data set used in the experiment is fb15k, and its statistical information is shown in Table 1. In addition, the training sample set is increased by constructing inverse relation.

| data set | relation | entity | Training set | Validation set | Test set |
|----------|----------|--------|--------------|----------------|---------|
| FB15K    | 1345     | 14951  | 483142       | 50000          | 59071   |

4.1. Parameter setting

In this experiment, both in the entity prediction and relationship prediction tasks are used Adam as the gradient optimizer, and its default parameter configuration is used: \( \beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8} \). All parameters in the model are regularized with L1 normal form, and dropout layer is used to prevent over fitting during the training.

The super parameters of D-Siamese include learning rate \( \lambda \), embedded vector dimension \( D \), mini batch = 6, penalty coefficient \( \lambda \) of regular term, dropout probability and success rate of negative sample sampling during training. We choose the value of learning rate form \{0.001, 0.005, 0.01, 0.1\}, and we also select the value of embedded vector dimension from \{50, 100, 150, 200\}, selecting the value of the penalty coefficient \( \lambda \) of the regular term from \{1e-5, 0.0001, 0.001\}. Value of mini batch \( B = 200 \). Select the dropout probability \( P_d \) from \{0.3, 0.5, 0.7\} and the negative sample sampling success rate \( p_y \) from \{0.25, 0.5, and 0.75\}. The parameter selection is determined according to the Mean rank value predicted by the validation set entity. Finally, the parameters of entity prediction are set as: \( l_r=0.01, b=200, \lambda=1e-5, p_b=0.3, d=100, p_r=0.25 \). Parameter setting of relationship prediction: \( l_r=0.01, b=200, \lambda=1e-5, p_b=0.3, d=50, p_r=0.5 \).

At the beginning of training, the embedded vectors of entities and relationships and the weight parameters of Dis-Siamese are initialized to \([-6/\sqrt{d}, 6/\sqrt{d}]\) is random numbers uniformly distributed in the range. In the task of entity prediction and relationship prediction, the maximum iteration of Dis-Siamese in the training process is 100, According to the TransE settings, the entity and relationship embedding vectors and model weight parameters are initialized to meet the uniform distribution random of \([-6/\sqrt{d}, 6/\sqrt{d}]\)
Table 2. Experimental results of entity prediction by DIS Siamese

| Metric                  | Mean Rank |         |         |
|-------------------------|-----------|---------|---------|
|                         | Raw       | Filter  | Raw     | Filter   |
| LFM                     | 285       | 171     | 24.0    | 32.2     |
| TransE                  | 245       | 124     | 43.6    | 45.8     |
| TransH                  | 190       | 82      | 47.9    | 67.2     |
| TransR                  | 198       | 77      | 48.3    | 68.1     |
| SME(bilinear)           | 284       | 156     | 32.4    | 42.6     |
| SME(linear)             | 276       | 154     | 30.9    | 40.8     |
| ProjE_pointwise         | 197       | 113     | 53.1    | 80.2     |
| ProjE_listwise          | 178       | 83      | 53.2    | 80.4     |
| ProjE_wlistwise         | 162       | 51      | 57.3    | 83.1     |
| Dis-Siamese_pointwise   | 201       | 110     | 54.1    | 80.6     |
| Dis-Siamese_listwise    | 173       | 82      | 57.1    | 77.2     |
| Dis-Siamese_wlistwise   | 161       | 51      | 58.3    | 83.7     |

4.2. Entity forecast and relationship forecast

In this section, the performance of Dis-Siamese is evaluated by entity prediction and relationship prediction on fb15k data set. Refer to transE and PTransE for specific experimental settings and model performance evaluation criteria. Among them, entity prediction aims to predict the missing head entity h or tail entity t in a given triple by sorting all entities in the data set. In the prediction of tail entities, the set of candidate triples is constructed by replacing the head entities in the given triples with the entities in the data set, and then the replaced entities are arranged in descending order according to the confidence score of the candidate triples. When predicting the head entity, it is equivalent to predicting the tail entity in the triple. The relationship prediction task is similar to entity prediction. It uses all the relationships in the data set to replace the relationships in each test triple to form a candidate triplet, and then arranges the replacement relationships in descending order according to the confidence score of the candidate triple. In this experiment, Mean Rank and Hits @k are also used as the evaluation criteria of the model performance. While, Mean Rank is used to measure the average ranking value of the correct entity or relationship, and Hits @ K is used to measure the percentage of the correct entity or relationship ranking in top-k. The filtered Mean Rank and Hits @ K ignore all other correct entities or relationships in the sorting results, and only focus on the sorting of the target entities or relationships in the test triples.

For example, if the target relationship between triples (Beijing, China) is (location in), and the relationship sorted as top-2 is capital of and locate in, the unfiltered original Mean Rank and Hits @ 1 values are 2 and 0, respectively. However, because the filtering operation ignores the correct relationship capital of, the filtered values of Mean Rank and Hits @ 1 are both 1. “Raw” and “Filter” are used to represent the original experimental results and filtered experimental results. In addition, this paper introduces weight to increase the importance of one-to-many, many-to-one and many-to-many complex relationships, namely Dis-Siamese_wlistise, and evaluates the performance of the model variants in experiments. The results of entity forecast and relationship forecast are shown in Table 2 and table 3 respectively. In this paper, ProjE and its benchmark model are used as comparison objects. Since the author of ProjE revised the experimental code after the paper was published, this paper lists the experimental results based on the revised code. Experimental results show that the performance of Dis-Siamese is better than the benchmark model in most cases. In addition, although softmax is mostly used for mutual exclusion and multi classification tasks and sigmoid can be used for entity relationship prediction and other non mutual exclusion classifications, the experimental results show that Dis-Siamese_listwise and Dis-Siamese_wlistwise achieve better performance than Dis-Siamese_pointwise. This is similar to ProjE's results.
Table 3. Experimental results of entity prediction by DIS Siamese

| Metric                  | Mean Rank |             |             |
|-------------------------|-----------|-------------|-------------|
|                         | Raw       | Filter      | Raw @1%     | Filter     |
| TransE                  | 2.9       | 2.6         | 63.6        | 85.8       |
| ProjE_pointwise         | 1.7       | 1.2         | 63.1        | 90.2       |
| ProjE_listwise          | 1.7       | 1.2         | 73.2        | 90.4       |
| ProjE_wlistwise         | 1.6       | 1.3         | 77.3        | 93.1       |
| Dis-Siamese_pointwise   | 1.6       | 1.3         | 74.1        | 90.6       |
| Dis-Siamese_listwise    | 1.5       | 1.2         | 77.1        | 93.2       |
| Dis-Siamese_wlistwise   | 1.5       | 1.2         | 78.3        | 93.7       |

Analysis of the experimental results: Dis-Siamese proposed in this paper uses Pseudo-Siamese network to transform (head entity, relation) and tail entity into the same feature space, and the similarity comparison between them is carried out in the same feature space. However, ProjE only fused the head entity with relationship, which is equivalent to the feature space transformation, and compared the fusion result with the tail entity without the feature space transformation. Therefore, the performance of Dis-Siamese is better than ProjE.

When sigmoid is applied to each of candidate entity, the ranking score of the positive candidate entity is larger than the negative candidate entity. However, point sorting cannot solve the problem of collaborative sorting of all candidate entities. In the list sorting, we can maximize the similarity between the binary mark vector and the sorting score vector, so that more positive candidate entities rank in front of negative candidate entities. Therefore, Dis-Siamese_listwise and Dis-Siamese_wlistwise can get better performance.

By increasing the weight of complex relations in the training process, it can improve the ability of the model to deal with complex relationship. Therefore, the performance of Dis-Siamese_wlistwise is better than Dis-Siamese_listwise.

5. Summary

According to the correspondence between the triple and the fact type simple question and answer, this paper proposes a distributed representation method of knowledge graph based on the Pseudo-Siamese network, which transforms the (head entity, relation) and the tail entity into the same feature space through the Pseudo-Siamese network, and takes the similarity of the two feature vectors in this space as the basis for calculating the confidence score of the input triple. In this paper, according to point sorting and list sorting respectively are used different training objective functions. By constructing inverse relation, that samples (tail entity, relation, head entity) are obtained for head entity prediction. At the same time, it improves the learning effect of the model by increasing the number of training samples.

References
[1] Zagoruyko S, Komodakis N. Learning to compare image patches via convolutional neural networks [A]. Proceedings of the IEEE conference on computer vision and pattern recognition [C], 2015: 4353-4361.
[2] Bertinetto L, Valmadre J, Henriques J F, et al. Fully-convolutional siamese networks for object tracking [A]. European conference on computer vision [C], Springer, Cham, 2016: 850-865.
[3] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines [A]. Proceedings of the 27th international conference on machine learning (ICML-10) [C], 2010: 807-814.
[4] Shi B, Weninger T. ProjE: Embedding ProjEction for knowledge graph completion [A].// Thirty-First AAAI Conference on Artificial Intelligence [C], 2017.
[5] Liu Q, Jiang H, Ling Z, et al. Probabilistic Reasoning via Deep Learning: Neural Association Models [J]. arXiv: Artificial Intelligence, 2016.