Attribute-Driven Capsule Network for Entity Relation Prediction

Jiayin Chen[1], Xiaolong Gong[1,2], Xi Chen[1(✉)], and Zhiyi Ma[1,3]

[1] Advanced Institute of Information Technology, Peking University, Hangzhou, China
{jychen,xlgong,xchen,mazhiyi}@aiit.org.cn
[2] Department of Computer Science, Shanghai Jiao Tong University, Shanghai, China
[3] School of Electronics Engineering and Computer Science, Peking University, Beijing, China

Abstract. Multi-attribute entity relation prediction is a novel data mining application about designing an intelligent system that supports inferencing across attributes information. However, most existing deep learning methods capture the inner structural information between different attributes are far more limited. In this paper, we propose an attribute-driven approach for entity relation prediction task based on capsule networks that have been shown to demonstrate good performance on relation mining. We develop a self-attention routing method to encapsulate multiple attributes semantic representation into relational semantic capsules and using dynamic routing method to generate class capsules for predicting relations. Due to the lack of multi-attribute entity relation data is a major obstacle in this task, we construct a new real-world multi-attribute entity relation dataset in this work. Experimental results show significant superiority of our model, as compared with other baselines.

Keywords: Entity relation prediction · Capsule networks · Self-attention routing

1 Introduction

Learning to predict relationship between the entities plays a vital role in recommendation system, knowledge base population and question answering system, etc. In the big data era, we aim to know the relation between two homogeneous or heterogeneous entities through the latent knowledge behind the data. Take an example in business mining field, we attempt to predict what the relationship between the two companies is, which can help an enterprise to search his potential customers or providers. The *relation prediction* (RP) task is different from *relation extraction* (RE) task, the entity pairs in RP task does not appear in the same semantic sentence and sometimes they may consist of many common attributes. For instance, there exists many common multiple attributes information in a pair of companies such as *company name, company address,*

H. W. Lauw et al. (Eds.): PAKDD 2020, LNAI 12084, pp. 675–686, 2020.
https://doi.org/10.1007/978-3-030-47426-3_52

company profile, business scope, etc. Hence, a demanding requirement for promoting RP is to develop a novel model that can support the entity relation for those multi-attribute data.

On RP task, most traditional methods are kernel-based method [1–4] that they need to do complicated feature engineering. Moreover, those methods limit to capture latent semantic information and they cannot extract some new effective features from relation examples easily. In recent years, a variety of neural network models [5–15] have been widely applied to relation prediction and achieved remarkable success. Those models are mainly based on an architecture of distributed representation, which learns a scalar-output to represent entity relation. All these methods can be divided into three categories: the CNN-based method [10,11], the RNN-based method [12,13] and the Transformer-based method [14,15]. Although above methods can capture latent semantic information due to the deep learning way, some disadvantages remain: 1) they heavily rely on the quality of entity semantic representation. Using one scalar-output to represent entity relation is limited because attribute information is abundant and diverse. 2) Above methods fail to retain the precise spatial relationships between high-level parts. The structural relationships such as the homogeneous information in all attributes are valuable. To address above problems, the capsule network methods have been proposed [16–18], which encapsulate multiple attributes into groups of neurons and replaces the scalar-output feature detectors with vector-output capsules to preserve additional information such as position and correlation [19]. Recently, capsule networks have achieved competitive results in classification tasks [18,20,21], relation extraction [22], especially for those structural information learning tasks [19,23]. In RP task, the entity relationship is usually correlate with their multiple attributes information. The capsule networks could represent those multiple attributes of entities as individual capsules, preserving the structural, correlation, relationship information between multiple attributes to drive entity relation prediction.

In this paper, we propose a novel Attribute-driven Capsule Network (**ACNet**) for entity relation prediction task, which retains the structural and correlation information by using capsule networks. Our **ACNet** makes attribute capsules generate relation capsules by developing a self-attention routing method, which assign weights to different attribute capsules and improve relation prediction performance. Furthermore, we adopt k-max pooling method to improve robust representation and training efficiency. The major contributions of this paper are briefly summarized as follows.

1. We propose a novel entity relation prediction approach based on capsule networks with self-attention routing method for those multi-attribute entity datasets. To the best of our knowledge, this is the first work that capsule network has been empirically apply on multi-attribute entity relation prediction.
2. We devise self-attention routing method between the attribute capsule layer and the relation capsule layer to improve the ability to capture relational semantic information. We conduct extensive experiments on a new real-world scenario CompanyRelationCollection (CRC) dataset that we constructed

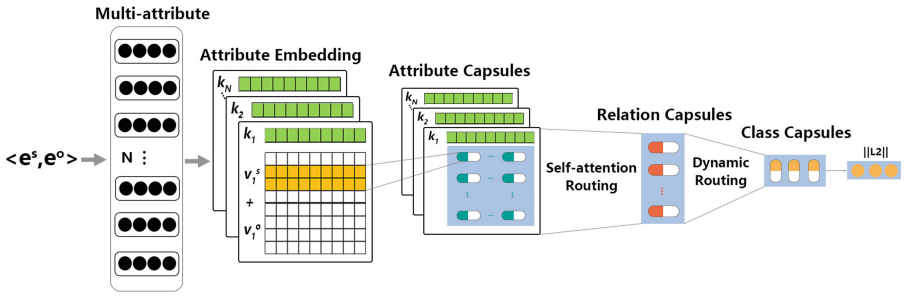

Fig. 1. The whole framework of **ACNet** method. It consists of four layers: 1) **Attribute Embedding Layer** converts each of common attributes into *attribute-key* and *attribute-value* vectors; 2) **Attribute Capsule** extracts feature from *attribute-value* vector and generates attribute capsules; 3) **Relation Capsule** uses self-attention routing method to aggregate attribute capsules and *attribute-key* vectors into a set of relation feature capsules; 4) **Class Capsule** produces classification capsules to represent each relation category.

from web and one public BlurbGenreCollection (BGC) dataset. The results demonstrate our **ACNet** consistently outperforms the state-of-the-art baselines.

The rest of the paper is organized as follows: Sect. 2 describes the proposed our model; Sect. 3 evaluates the approach; Sect. 4 concludes the paper.

2 Model

2.1 Definitions

Definition 1 (Relation prediction). *Let* $D=\{e^1, e^2, \ldots e^n\}$ *represent a set of entities, where* $|D| = n$*. The ordered pair* $<e^s, e^o>$ *in* D *denotes two entities has a relevant link represented by* $r<e^s, e^o>$*, which is called relation. The learning task is to improve the target predictive function for relation* $f_r<e^s, e^o>$*.*

Definition 2 (Attribute information). *Let* $e^s = \{k_1 : v_1^s, \ldots, k_N : v_N^s\}$ *represent multi-attribute entity information, where* $|e^s| = N$ *means that there exists* N *attributes for each entity. And multi-attribute in the form of key-value* $(k : v)$ *pairs to indicate their information. We utilize* E^k *(attribute-key) and* E^v *(attribute-value) to represent the key and value information of one attribute, respectively.*

2.2 Attribute Embedding Layer

The whole framework of our method is shown in Fig. 1. The purpose of the attribute embedding layer is to turn each common attribute into two separate embedding vectors. Let $M^E \in \mathbb{R}^{d_w \times |V|}$ represent attribute information embedding matrix, where d_w is the dimension of word vectors and $|V|$ is the vocabulary size. Here, the embedding for information of j-th attribute can be separated into two vectors, using $E^{k_j} \in \mathbb{R}^{1 \times d_w}$ to represent *attribute-key* embedding vector and $E^{v_j} = \{x_1, \ldots, x_i, \ldots, x_L\} \in \mathbb{R}^{L \times d_w}$ to represent *attribute-value* embedding vector, where x_i is the i-th word in the *attribute-value* sentence and L is the sentence length. For a relation pair $<e^s, e^o>$, due to the relationship is strongly related to their common attributes information, we concatenate their *attribute-value* vectors for each attribute and they can be formulated as $<e^s, e^o> = \{E^{k_1} : E^{v_1^s} + E^{v_1^o}, \ldots, E^{k_N} : E^{v_N^s} + E^{v_N^o}\} = \{E^{k_1} : E^{v_1}, \ldots, E^{k_N} : E^{v_N}\}$.

2.3 Attribute Capsule Layer

This layer aims to use multiple convolution operations to extract n-gram features from the *attribute-value embedding*, which contain local semantic information about the *attribute-key* in a fixed window. In general, let $x_{i:i+j}$ refers to the concatenation of words $x_i, x_{i+1}, \ldots, x_{i+j}$. Multiple convolution operations involves a kernel group $F \in \mathbb{R}^{d_p \times (d_w \times h)}$, which is applied to a window with h words to generate multiple h-gram features. $d_w \times h$ is the size of one convolutional kernel, h is the n-gram size and d_p is the dimension of one attribute capsule. For a j-th attribute, a feature c_i is produced from a window of words $x_{i:i+h-1}$ by

$$c_i = F \odot E^{v_j}_{i:i+h-1} + b \tag{1}$$

where $\odot$ denotes the component-wise multiplication and $b \in \mathbb{R}$ is a bias term. Thus, we can get a set of attribute capsules $c \in \mathbb{R}^{d_p \times (L-h+1)}$, which encapsulates n-gram features and extracts from the whole textual information of an *attribute-value*.

In fact, different kernel groups F can capture different categories of semantic meaning. We repeat the above procedure B times with different kernel groups, and get multiple channels of features to represent B categories of semantic meaning. The final output of this layer is arranged as $C \in \mathbb{R}^{B \times d_p \times (L-h+1)}$, where $C = [c_1, c_2, \ldots c_B]$.

2.4 Relation Capsule Layer

Self-Attention Routing Approach. For a common attribute, different n-gram features are extracted from the *attribute-value* and they have different effects on the *attribute-key*. What's more, n-gram features contain different semantic information of entity relation. To alleviate aforementioned problem, and following self-attention mechanism [24,25], we propose a novel attention routing approach to compute the weight for the n-gram features of h-size window in E^{v_j}.

First, we apply a fusing convolution operation to the embedding E^{v_j} with a kernel $V_s \in \mathbb{R}^{d_w \times h}$, and getting a set of feature vectors $F_s \in \mathbb{R}^{1\times(L-h+1)}$. Second, we let E^{k_j} as our query input, a simple linear projection is used to construct the query $Q = E^{k_j}W^q$, and the key $K = F_sW^k$. Then, we use the query Q to perform a scaled dot-product attention over K. The returned scalars can be put through a softmax function to produce a set of weights.

$$a_i = softmax\left(\frac{E^{k_j}W^q(F_sW^k)^T}{\sqrt{d_w}}\right) \tag{2}$$

where $a_i \in \mathbb{R}^{1\times(L-h+1)}$, $W^q \in \mathbb{R}^{d_w\times u}$ and $W^k \in \mathbb{R}^{(L-h+1)\times u}$ represent weighted parameter matrices. The generate attention weight $a_i \in [0, 1]$ contains information with respect to *attribute-value*. It controls how much information in current n-gram feature can be transmitted to the next layer. If a_i is zero, the feature capsule would be totally blocked. Since produced B different channels of attribute capsules in above layer, we repeat the above computational process B times to get the whole attention routing weights $A \in \mathbb{R}^{B\times1\times(L-h+1)}$, where $A = [a_1, a_2, \ldots, a_B]$. Finally, the attribute capsules are routed using these weights:

$$S = C \odot A \tag{3}$$

where $S \in \mathbb{R}^{B\times d_p\times(L-h+1)}$ is the attribute-customized feature capsules, and $\odot$ denotes element-wise multiplication.

Relation Capsule Generation. The above generated S is transformed from attribute capsules. Though encoding key-related information, there are still many unrelated capsules in S. Moreover, the large number of capsules in S may prevent the next layer from learning robust representations. If using a method similar to the max pooling, it will lead to lose some capsules about the structural or position information. Hence, we adopt a compromise method, k-max method in S to aggregate all attribute capsules in the third channel horizontally.

$$P = max(k, [S_1, S_2, .., S_{L-h+1}]) \tag{4}$$

where $P \in \mathbb{R}^{B\times k\times d_p}$, k is a constant that meaning to preserve top k capsules and *max* is a descending sort operation. Through Eq. 4, the network can filter out many unimportant capsules and obtain more relation capsules.

From Eq. 1 to Eq. 4, a attribute can generate a set of capsules P and we loop N attributes in the same way to get a set of all attributes capsules u. Then we combine all attributes capsules together to produce relation capsules:

$$u = [P_1, P_2, ..., P_N] \tag{5}$$

where $u \in \mathbb{R}^{(B\times K\times N)\times d_p}$. Then, we use the non-linear "squash" function [16], using the length of each relation feature capsule u_i to represent the probability that u_i's feature meaning is present in the current input.

$$u_i \leftarrow \frac{\|u_i\|^2}{1+\|u_i\|^2}\frac{u_i}{\|u_i\|} \tag{6}$$

Algorithm 1. Dynamic Routing Algorithm

Procedure ROUTING($\hat{u}_{j|i}, r, l$) ;
for all capsule i in layer l *and capsule* j *in layer (l+1):* $b_{ij} \leftarrow 0$.
for r *iterations* ***do***
    for all capsule i *in layer* l: $c_i \leftarrow softmax(b_i)$
    for all capsule j *in layer (l+1):* $s_j \leftarrow \sum_i c_{ij}\hat{u}_{j|i}$
    for all capsule j *in layer (l+1):* $v_j \leftarrow squash(s_j)$
    for all capsule i *in layer* l *and capsule* j *in layer (l+1):* $b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i}.v_j$
end forreturn v_j;

2.5 Class Capsules Layer

In the capsule network, it uses class capsules to represent entity relation categories that means the number of relation capsules to be consistent with categories of entity relations. Let the number of relation categories be m, and there are m relation capsules learned in this layer. Each relation capsule is used for calculating the classification probability of each relation in entity prediction task. Hence each class capsule should have its own routing weights to adaptively aggregate relation capsules from the previous layer.

A capsule's prediction vector $\tilde{u}_{j|i}$ is generated by multiplying the output u_i by a weight matrix W_{ij}, where $W_{ij} \in \mathbb{R}^{d_r \times d_p}$ is a weight matrix, d_r and d_p are the dimensions of class capsule j and relation capsule i, u_i is the vector representation of relation capsule i.

$$\hat{u}_{j|i} = W_{ij}u_i \tag{7}$$

Then all prediction vectors generated by relation capsules are summed up with weights c_{ij} to obtain the vector representation s_j of class capsule j:

$$\mathbf{s_j} = \sum_i c_{ij}\tilde{u}_{j|i} \tag{8}$$

where c_{ij} is a coupling coefficient that determine the contribution of each relation capsule's output to a class capsule are calculated using a dynamic routing heuristic [16] and defined by a "routing softmax":

$$c_{ij} = \frac{exp(b_{ij})}{\sum_k exp(b_{ik})} \tag{9}$$

where each b_{ij} is the log prior probability that a relation capsule i should pass to a class capsule j. It is computed using a dynamic routing approach which is written in Algorithm 1 [16].

After that, we apply the non-linear "squash" function again to $\mathbf{s_j}$ in Eq. 6 to get a final representation $\mathbf{v_j}$ for class capsule j.

$$\mathbf{v_j} = squash(\mathbf{s_j}) \tag{10}$$

2.6 Margin Loss

We use the length of a class vector to represent the probability of the relationship between entities. The capsule length of the active relation should be larger than others. We adopt a separate margin loss L_j for each class capsule j in our task:

$$L_j = T_j max(0, m^+ - \|\mathbf{v_j}\|)^2 + \lambda(1 - T_j) max(0, \|\mathbf{v_j}\| - m^-)^2 \quad (11)$$

where $T_j = 1$ if the entity relation is present in class capsule j. We set m^+=0.9, and $m^- = 0.1$, $\lambda = 0.5$ following in the research [16]. The total loss is simply the sum of the losses of all class capsules, $L_T = \sum_{j=1}^{J} L_j$.

3 Experiments

3.1 Datasets

We evaluate our model on two datasets[1]: **CompanyRelationCollection (CRC)** refers to company entities and **BlurbGenreCollection (BGC)** refers to book entities. The **CRC** is a Chinese company entity dataset that collected from the Internet. It consists of 58,013 company entities and there are 6 common attributes for each company entity, which includes *company_name, company_address, company_type, industry_category, business_scope* and *abstract*. For a pair of company entities, there exists three major links content, which includes customer (C), provider (P), rival (R). Figure 2 gives an example of a relationship in **CRC** dataset, which demonstrates *Entity_O* is a customer of *Entity_S*. In our task, we retain only one relationship between two company entities. The **BGC** is an English public dataset [23]. It includes 91,892 book entities and each entity has 3 common attributes. We define three kinds of link according to book categories, which includes similar(S), presumably-similar(P), dissimilar(D). Table 1 lists some important quantitative characteristics of both datasets.

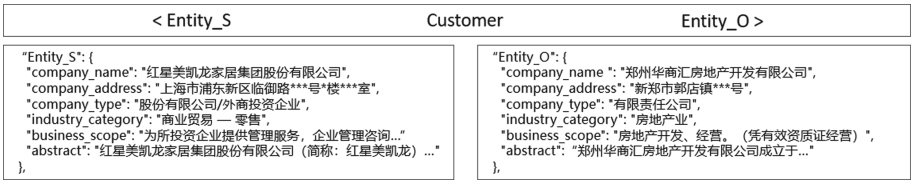

Fig. 2. An example of one relationship in CRC dataset.

[1] The dataset is available at https://github.com/ACNet.

Table 1. Quantitative characteristics of both datasets

	CRC	BGC
Number of entities	58,013	91,892
Number of attributes per entity	6	3
Total number of relationships	3(C,P,R)	3(S,P,D)
Number of relational pairs	61,441	918,920
Train set	43,009	735,136
Validation set	9216	91,892
Test set	9216	91,892

3.2 Experimental Setting

Due to different text lengths of *attribute-values*, we fix text length of all *attribute-values* equals to 200. We train words embedding [26] in two dataset respectively.

For all the experiments below, we set some parameters: $d_w = 200$ for word embedding dimensionality, $d_p = 16$ and $d_r = 24$ for the dimension of a relation capsule and a class capsule, respectively. We tune some parameters of our models by grid searching on the validation dataset and grid search is utilized to select optimal learning rate λ for Adam optimizer, we final set our λ equal to 0.001 and filters number B equals to 64. The significant parameter k with 12 different values in our experiment and detailed discussion is presented in Sect. 3.5. Table 2 shows that hyper-parameters of the optimum model were selected by the evaluation results on the validation dataset. For other parameters, we use empirical settings because they make few influence on performance of our model. Three widespread used evaluation metrics are applied in the experiments, which includes the precision, recall and F1. We select F1 as our final main evaluation indicator.

3.3 Baselines

To demonstrate the superiority of **ACNet** on relation prediction task between multi-attribute entities, we compare it with following six baselines: **CNN** proposes a convolutional deep neural network model for entity relation mining. **PCNN** puts forward a piecewise CNN model for distant supervision for relation mining. **BLSTM** proposes a bidirectional LSTM model for relation mining. **ATT-BLSTM** is a bidirectional LSTM model with attention mechanism. **BERT** is a pre-training bidirectional transformer model for relation mining. All above methods are based on the idea of classification and they are suitable for our relation prediction tasks. **Basic-Caps** is the original capsule network model without self-attention routing mechanism.

Table 2. List of hyper-parameters

Hyper-parameter	Value
Word embedding size (d_w)	200
Dimension of attribute capsule (d_p)	16
convolutional kernel size (h)	2
Number of convolution kernels (B)	64
Self-attention matrix parameter (u)	100
Number of reserved top capsules (k)	10
Dimension of class capsule (d_r)	24
Learning rate (λ)	0.001

Table 3. The results of Comparison of different methods. Best scores are in bold.

Method	CRC			BGC		
	Precision	Recall	F1	Precision	Recall	F1
CNN [10]	0.7706	0.7012	0.7343	0.8420	0.8265	0.8342
PCNN [11]	0.7825	0.7103	0.7447	0.8578	0.8299	0.8436
BLSTM [12]	0.7682	0.7066	0.7361	0.85378	0.8122	0.8324
BERT [27]	**0.8067**	0.6936	0.7459	**0.8628**	0.8345	0.8484
Basic-Caps	0.7528	0.7331	0.7428	0.8534	0.8304	0.8417
ACNet	0.7662	**0.7405**	**0.7531**	0.8612	**0.8405**	**0.8507**

3.4 Main Results

We conduct experiments on the seven methods mentioned above. The comparison results of all models are shown in Table 3. It is clear that our attribute-driven capsule network achieves the highest F1 scores, outperforming the other deep network-based methods, simultaneously in terms of recall. Our **ACNnet** has the smallest fluctuation between precision and recall on two datasets. Both situations demonstrate that our method is superior to other models on relation prediction between multi-attribute entities and RP task learning is more robust.

The results of **ACNet** significantly outperform the **Basic-Caps** and others on the F1 indicator show that the self-attention routing approach is effective, which indicates that our approach could capture more relational semantic information and enhance the ability to represent class capsules by self-attention mechanism. Moreover, capsule networks have shown to identify and combine relational information with common attributes more accurately than the baselines.

BERT performs the best result on precision indicator among all baselines except our model, which demonstrates that the powerful pre-training model can achieve richful representation information. We also can observe that the self-attention routing approach beneficial to improve relation prediction between

multi-attribute entity. Among CNN-based methods, **PCNN** achieves higher performance than **CNN** since it uses dynamic max pooling method and preserves some level of information about attributes. However, **BLSTM** is better than **ATT-BLSTM** beyond our expectations. **ATT-BLSTM** performs the worst among all others, as maybe its based attention model can not extract more effective attribute features.

3.5 Parameter Analysis

Stability of Training. In order to investigate the difference between our model and other models in training process, we conduct a statistical analysis on training loss where we set epochs equal to 10 and select the lowest training loss at each epoch for comparison. The results are shown in Fig. 3. We can observe that the capsule networks convergence much faster than other methods, getting a stable training process and achieve less fluctuation. The reason behind this situation should be that capsule network discard pooling and retain all the feature information.

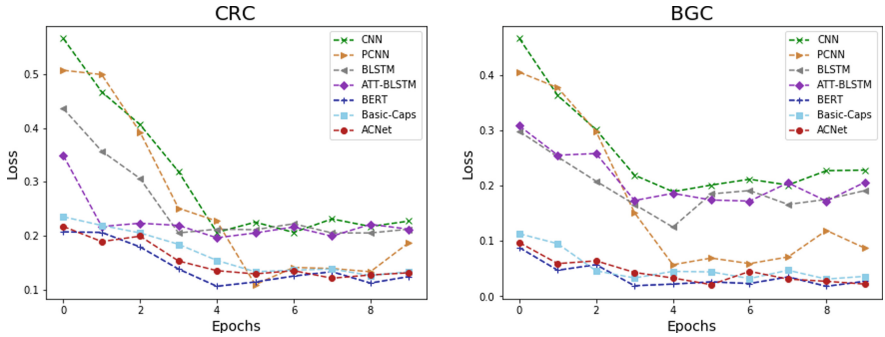

Fig. 3. The result of training loss from all models on two datasets.

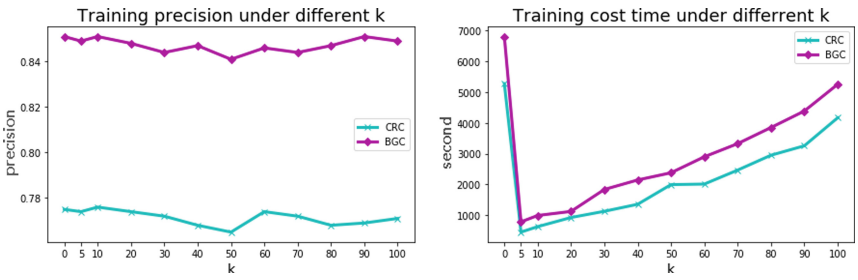

Fig. 4. Training results under different k values

k-max Pooling. To explore whether the capsule networks discard many features by the pooling way will have an impact on the prediction results, we use k-max pooling method in **ACNnet** to test it. We set 12 different k values and $k = 0$ means without using pooling. The results are shown in Fig. 4. The test results show that k-max pooling method does not decrease the precision of entity relationship prediction. Conversely, using a smaller k can greatly reduce training time and increase training efficiency. This also illustrates that it is effective to retain the main attribute capsules and filter out many unimportant capsules.

4 Conclusion

We present attribute-driven capsule network model for relation prediction between multi-attribute entities in this work. In order to capture relational information from common attributes and improve relation prediction, we develop self-attention routing method based on capsule networks to generate relational capsules and link with class capsules. The experimental results demonstrate the effectiveness of our model on RP task. Our future work includes: (i) to enrich the representation of the capsule network with more attributes, (ii) to consider entity relation prediction with different attributes, (iii) to apply some methods of the recommended domain on entity relation prediction.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (No. 61672046). We thank all the anonymous reviewers for their insightful comments.

References

1. Mooney, R.J., Bunescu, R.C.: Subsequence kernels for relation extraction (2005)
2. Bunescu, R.C., Mooney, R.J.: A shortest path dependency kernel for relation extraction (2005)
3. Yu, K., Chu, W., Yu, S., Tresp, V., Xu, Z.: Stochastic relational models for discriminative link prediction. In: Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, 4–7 December 2006 (2006)
4. Li, J., Zhang, Z., Li, X., Chen, H.: Kernel-based learning for biomedical relation extraction. J. Am. Soc. Inf. Sci. Technol. **59**(5), 756–769 (2008)
5. Nguyen, T.H., Grishman, R.: Relation extraction: perspective from convolutional neural networks. In: Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing, pp. 39–48 (2015)
6. Li, Z., Ding, N., Liu, Z., Zheng, H., Shen, Y.: Chinese relation extraction with multi-grained information and external linguistic knowledge. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 4377–4386 (2019)
7. Sahu, S.K., Christopoulou, F., Miwa, M., Ananiadou, S.: Inter-sentence relation extraction with document-level graph convolutional neural network. arXiv preprint arXiv:1906.04684 (2019)

8. Miwa, M., Bansal, M.: End-to-end relation extraction using LSTMs on sequences and tree structures. arXiv preprint arXiv:1601.00770 (2016)
9. Lin, Y., Shen, S., Liu, Z., Luan, H., Sun, M.: Neural relation extraction with selective attention over instances. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 2124–2133 (2016)
10. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J., et al.: Relation classification via convolutional deep neural network (2014)
11. Zeng, D., Liu, K., Chen, Y., Zhao, J.: Distant supervision for relation extraction via piecewise convolutional neural networks. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1753–1762 (2015)
12. Zhang, D., Wang, D.: Relation classification via recurrent neural network. arXiv preprint arXiv:1508.01006 (2015)
13. Zhou, P., et al.: Attention-based bidirectional long short-term memory networks for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 207–212 (2016)
14. Papanikolaou, Y., Roberts, I., Pierleoni, A.: Deep bidirectional transformers for relation extraction without supervision. arXiv preprint arXiv:1911.00313 (2019)
15. Wang, H., et al.: Extracting multiple-relations in one-pass with pre-trained transformers. arXiv preprint arXiv:1902.01030 (2019)
16. Sabour, S., Frosst, N., Hinton, G.E.: Dynamic routing between capsules. In: Advances in Neural Information Processing Systems, pp. 3856–3866 (2017)
17. Hinton, G.E., Sabour, S., Frosst, N.: Matrix capsules with EM routing (2018)
18. Zhao, W., Ye, J., Yang, M., Lei, Z., Zhang, S., Zhao, Z.: Investigating capsule networks with dynamic routing for text classification. arXiv preprint arXiv:1804.00538 (2018)
19. Chen, Z., Qian, T.: Transfer capsule network for aspect level sentiment classification. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 547–556 (2019)
20. Xi, E., Bing, S., Jin, Y.: Capsule network performance on complex data. arXiv preprint arXiv:1712.03480 (2017)
21. Kim, J., Jang, S., Park, E., Choi, S.: Text classification using capsules. Neurocomputing **376**, 214–221 (2020)
22. Zhang, N., Deng, S., Sun, Z., Chen, X., Zhang, W., Chen, H.: Attention-based capsule networks with dynamic routing for relation extraction. arXiv preprint arXiv:1812.11321 (2018)
23. Aly, R., Remus, S., Biemann, C.: Hierarchical multi-label classification of text with capsule networks. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pp. 323–330 (2019)
24. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
25. Santoro, A., et al.: Relational recurrent neural networks. In: Advances in Neural Information Processing Systems, pp. 7299–7310 (2018)
26. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
27. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)