ISSN 1870-4069

Event-Argument Linking in Hindi for Information Extraction in Disaster Domain

Sovan Kumar Sahoo, Saumajit Saha, Asif Ekbal,
Pushpak Bhattacharyya, Jimson Mathew

Indian Institute of Technology Patna,
Department of Computer Science and Engineering,
India

{sovan.pcs17, saumajit.mtmc17, asif, pb, jimson}@iitp.ac.in

Abstract. Event extraction is an important task in Natural Language Processing. Extracting event triggers and arguments from text is a very important sub-task of information extraction. If a sentence contains only a single event and one or more arguments, then it is obvious to assume that all the arguments are linked to that particular event. However, when a single sentence consists of multiple events and arguments, we need to link arguments with their respective events. In this paper, we develop a deep learning based approach to solve the problem of event-argument linking. We construct the task as a problem of classification, where for a given pair of event and candidate argument, the system has to decide whether they are linked to each other or not. As there is no available data in Hindi, we crawl the news data from different sources, annotate them following proper guidelines, and create a benchmark setup for event-argument linking. We believe that this is the very first attempt for event-argument linking in Hindi[1].

Keywords: Event-argument linking, deep learning, Hindi.

1 Introduction

Nowadays due to the advancement of electronic media, a massive amount of digital contents is uploaded very frequently on the internet. Extracting relevant information manually from this vast data is impossible. Information extraction concerns with developing the tools and techniques to mine the most relevant information from these data.

Event extraction is a crucial task of information extraction, used to detect the occurrence of an event along with its other details such as the time, place, agent, intensity and so on. Event mention refers to any phrase or event which describes an event. It also includes triggers and arguments. Event trigger points out the main word which highlights the occurrence of an event. Argument of an event refers to the attributes (describing the event) such as the location of occurrence of the event, time of occurrence of the event, participants involved and so on. Detection of event trigger, classification of

[1] https://github.com/Saumajit/EAL

event trigger, extraction of argument, and event-argument linking are the four important components of a typical information extraction system.

The fourth one, i.e., event-argument linking is more complex compared to the first three tasks. Generally, if any sentence consists of only one event and multiple arguments, then we can assume that all the arguments are linked to that particular event. However, if a particular sentence consists of multiple events and multiple arguments then it is difficult to decide which arguments are linked to which events. Though the task of information extraction has been explored significantly for the resource-rich language like English, this has not been the case with resource-poor language like Hindi.

One reason is the lack of availability of the annotated data for the target tasks- be it detection of event trigger, classification of event trigger, extraction of argument or event-argument linking. In our current work, we present an effective deep learning approach for event-argument linking for Hindi. We design the task of event-argument linking as a classification problem, where for a given pair of event and candidate argument, the system predicts whether they are linked to each other or not.

We use Convolutional Neural Network (CNN) [7] as feature extractor and try to classify whether there exists a relationship between an event and an argument or not. We also observe that event can lie either to the left or to the right of an argument in a sentence. Thus there exists a bidirectional relationship between an event and its arguments in a sentence. We, therefore, use Bidirectional Long Short-Term Memory (Bi-LSTM) [12] followed by CNN to address this bidirectional relationship.

In our experiment, we use Hindi news data from disaster domain. The reason behind choosing this domain is its importance and impact in our society. Extracting disaster-related information from news documents as well as from the other sources is crucial. It is useful to spread awareness among citizens and to provide relevant information to the other stakeholders such as the government departments and humanitarian agencies.

This information not only makes everyone alert but helps in overall disaster management. There is no existing dataset for information extraction in Hindi. We crawl news data from various newspapers and annotate them for our particular task. We believe that this is the very first attempt for event-argument linking in Hindi.

1.1 Problem Definition and Contributions

Given a Hindi sentence comprising of the sequence, w_1, w_2, e_1, e_2,..., e_i, w_3,...,w_k, a_1, a_2,..., a_j, w_{k+1},..., w_n, where e_i is known as an event trigger and a_j is known as a candidate argument, the task is to predict whether there exists a relationship between an event trigger e_i and an argument trigger a_j or not. Let us consider an example sentence which consists of two events and four arguments. Here, we have a total of eight event-argument pairs.

As the place argument कुलगाम (Kulgam) is linked to the event trigger बम विस्फोट (Bomb blast), we assign the classification label as '1', whereas the argument कुलगाम (Kulgam) is not linked with the event trigger आत्मघाती हमले (Suicide attack), so the classification label, in this case, is '0'. Table 1 depicts the possible event-argument pairs for the given example.

Table 1. Training instances generated from the sentence given in the above example.

Event-Argument pair	Classification label
बम विस्फोट, एक नागरिक सहित चार लोग	1
बम विस्फोट, कुलगाम	1
बम विस्फोट, छह लोग	0
बम विस्फोट, सोपियन जिले	0
आत्मघाती हमले, एक नागरिक सहित चार लोग	0
आत्मघाती हमले, कुलगाम	0
आत्मघाती हमले, छह लोग	1
आत्मघाती हमले, सोपियन जिले	1

– **Input Hindi Sentence :** कुलगाम में एक बम विस्फोट में एक नागरिक सहित चार लोग मारे गए हैं जबकि सोपियन जिले में एक आत्मघाती हमले में छह लोग मारे गए हैं ।
– **Transliteration :** Kulagaam mein ek bam visphot mein ek naagarik sahit chaar log maare gae hain jabaki sopiyan jile mein ek aatmaghaatee hamale mein chhah log maare gae hain.
– **Translation :** Four people, including a civilian, were killed in a bomb blast in Kulgam, while six people were killed in a suicide attack in the Sopiyan district.

The contribution of our current research is two-fold, *viz.* (i). We propose a deep learning based event-argument linking system in Hindi for disaster domain; and (ii). Provide a benchmark setup for event-argument linking in Hindi language.

2 Related Work

In our current work, we focus on finding the relation between an event and its corresponding argument using deep neural networks. Thus our current work falls under the lines of research of neural relation extraction. Relation extraction using deep learning technique has already been explored by the research community [16,11,14,9,13,8,10,17,15,18,2,19,6,4]. Convolutional Neural Network(CNN) is long established in relation classification. [16] suggested a CNN based relation classification approach for the first time where CNN was used to extract sentence level feature.

The features of CNN were extracted by taking all the tokens of the sentence as input, where each token was represented as the concatenation of word feature and position feature. The authors also extracted lexical level features like word embeddings of marked nouns and their context tokens and WordNet hypernyms. Both the features were then concatenated into a single vector which was then passed into *Softmax* classifier for classification. After the success of CNN in relation classification, [11] proposed a CNN based approach that performs classification by Ranking CNN (CR-CNN).

They used a novel pairwise ranking loss function that helped to diminish the impact of artificial class *Other*. [14] proposed a robust model that learns from the shortest dependency paths through a CNN. They also suggested a negative sampling strategy into their CNN model to handle relation directionality. The advantage of multiple window sizes for convolutional filters was used in [9]. Thus, their model allows the network

to capture wider ranges of *n*-grams. They also used position embedding features. A multilevel attention CNN was proposed in [13].

They used primary attention at the input level to capture entity-specific attention and secondary attention with respect to target relation for relation specific pooling attention. They claimed that their novel mechanism allows their model to detect more subtle cues of the input sentences despite their heterogeneous structure. They also introduced in this paper a novel pair-wise margin-based objective function.

Though CNN-based methods can capture high-level features, they overlooked the hierarchical and syntactical information of the input sentence. Based on this observation, the authors in [8] introduced the hierarchical layers and dependency embedding to CNN based methods to capture both the hierarchical feature and dependency structure in the window size. In a very recent work [10], a CNN based model with adversarial training method was proposed. Though CNN is very successful in capturing features for relation extraction, it captures local feature and fails to take into consideration the long-distance dependency between the nominal pairs.

To deal with this issue, the authors in [17] presented a framework based on Recurrent Neural Network (RNN). A novel neural network SDP-LSTM was proposed in [15] where they picked heterogeneous information along Shortest Dependency Path (SDP) using four different information channels. They introduced a Long Short-Term Memory (LSTM) which was built upon dependency path. To take the directionality of relation into consideration, they separated an SDP into two sub-paths where each path was from an entity to the common ancestor node.

In recent work in [18], the authors used the attention layer and tensor layer on the top of Bi-LSTM to capture word level context information and complex connection between two entities.

So far it is seen that both the CNN and RNN have been used to extract the relations. However, some researchers used the combination of both the neural network architectures to capture both the local features as well as the long distance relationship between the two entities. For example in [2], the authors used CNN on the top of LSTM units which picked up necessary information along SDP and inverse SDP at the same time through two separate channels.

In another work reported in [19], the authors used the combination of CNN and RNN along with an attention layer in between them. Apart from this, some other approaches are also reported in the literature.

In [6], authors tried to use syntax information of sentences to model the entities. They proposed to learn syntax-aware entity embeddings based on tree-GRU. They first encoded the context of entities on a dependency tree in sentence-level. Then both inter-sentence and intra-sentence attentions were used to obtain sentence set-level entity embeddings over all the sentences which contain the focused entity pair. Finally, this entity embedding combined along with a CNN based sentence embedding was used for relation extraction.

Reinforcement learning was used in [4] to deal with the noisy labeling problem in distant supervision based relation extraction methods. Their model has two modules *viz.* instance selector and relation classifier. The instance selector uses reinforcement

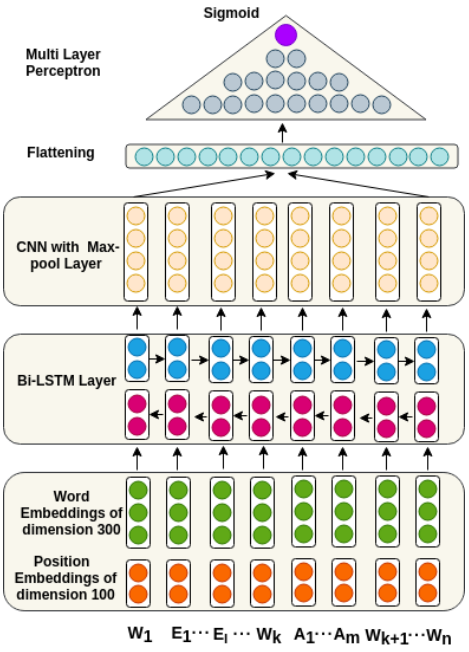

Fig. 1. The architecture of the proposed model. Here W_i denotes the words of the input sentence. E_i and A_i denote the event and argument trigger, respectively.

learning to choose the high-quality sentences and feeds the relation classifier which eventually makes the prediction and provides rewards to the instance selector.

3 Methodology

In this section, we describe the approach that we have followed for event argument linking.

3.1 Architecture

Figure 1 depicts the overall system diagram that we have used for event-argument linking. The network takes vector representation of each word of the input sentence as input and passes the vectors to a Bi-LSTM layer which captures the long term relationship between the event-argument from both the directions. The output of the Bi-LSTM layer is passed through a single-layered CNN. CNN tries to extract local convoluted features. The output of CNN is then fed into a multi-layer perceptron (MLP) model followed by a *Sigmoid* activation function for binary classification.

3.2 Input Representation

Each word w_i of the input sentence $S_i = (w_1, w_2, \ldots, w_n)$ is represented by the concatenation of two types of embeddings: (i) word embeddings (WE) which capture

Words	Four	people	including	a	civilian	were	killed	in	a	bomb	blast	in	Kulgam	while	six	people	were	killed	in	a	suicide	attack	in	Sopian	district
Relative Position w.r.t. Event	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Relative Position w.r.t. Arguments	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12

Fig. 2. Representation of the input sentence. Here each word has two relative positions with respect to Event and Argument respectively.

syntactic and semantic meaning of the word; (ii) a position embedding (PE) which identifies both the target event and arguments of our interest.

The PE also identifies the proximity of each word with respect to the target event and argument words or phrases. The input sentence S_i is the sequence of vectors $S_i = (w_1, w_2, \ldots, w_n)$, where $w_i \in R^d$ and $d = d_w + 2d_p$. d_w and d_p are the dimensions of word embedding and position embedding respectively. We choose the maximum length of each input sentence to be 100. We, therefore, use zero padding for shorter sentences and truncate the longer sentences.

3.3 Word Embedding

For word embedding (WE) of each word, we use pre-trained *fastText* [5] word vectors. These embeddings were trained on Hindi Common Crawl and Wikipedia dataset. The size of the word embedding used in our experiments is 300. The pre-trained word-embedddings are downloaded from *fastText* website[2].

3.4 Position Embedding

Position embedding (PE) was successfully applied in [16] for relation extraction. For position embedding of each word, we at first calculate the relative distance of each word with respect to event and argument trigger respectively. The relative distance can be both positive and negative. Each distance is then represented by a random vector of dimension 50.

4 Datasets and Experiments

Here in this section, we provide a description of the dataset that we have prepared for our experiments, report the results and then provide a useful analysis.

4.1 Dataset

As there was no existing dataset for event-argument linking in Hindi, we have prepared it by ourselves. The news data related to disaster events are crawled from the different news portals. All the articles are converted into XML formats and then annotated with event triggers, arguments and for event-argument linking. We have followed TAC KBP [3] annotation guidelines for our annotation task. Three annotators, with a good linguistic background, were employed for the annotation task.

Table 2. Dataset statistics. Here 'relevant instances' refer to the no of event-argument links.

Number of XML files used for training	824
Total number of relevant instances in the training dataset	7554
Number of XML files used for testing	194
Total number of relevant instances in the testing dataset	1934

Table 3. Hyperparameters used in our experiments.

Hyper parameters	**# of Epochs**	**Dropout**	**Batch size**	**# of filters**	**# of dense layer neurons**	**Dimension of WE** (d_w)	**Dimension of PE** (d_w)
Value	100	0.5	64	64	100	300	50

The tag set representing events is organized into an ontology which includes two types of events - *Natural* and *Man-made*, and ten types of arguments - *Place*, *Time*, *Casualty*, *Reason*, *Type*, *Participant*, *Intensity*, *Magnitude*, *Name* and *Speed*. The ontology has three levels where both *Natural* and *Man-made* disaster types are further divided into different sub-types. We have a total of 29 sub-types of disasters in our Hindi corpus. We measure the inter-annotator agreement ratio by asking all the three annotators to annotate 5% of total documents. The multi-rater Kappa agreement ratio of 0.85 was observed.

Table 2 shows the train-test split of total Hindi dataset. The annotated sentences are then used as input to our classification problem. Let us assume that a sentence contains two events and three arguments. We create six instances by considering each of the six event-argument pairs. For each such instance containing a particular event-argument pair, the relative distance for each word with respect to that event and argument changes. We assign the label as binary-valued (1 or 0) indicating the presence or absence of linkage between the event and argument.

4.2 Experimental Setup

For developing the system, we use the Python-based Keras [3] library with TensorFlow [1] backend. The hyperparameters are shown in Table 3.

4.3 Results and Analysis

Table 4 reports the results of all the different models in terms of *Precision*, *Recall* and *F1-Score*. Figure 3 shows the number of correctly predicted instances for both the classes for different architectures. For example, Model 3 has predicted YES (label=1) for 1068 instances where the actual label for all these instances were YES (label=1). Similarly, it has predicted NO (label=0) for 228 instances where the actual label for all these instances were NO (label=0).

Figure 3 also shows that our proposed model (Model 3) performs better than all the other models for both the classes even though *F1-score* is slightly lesser for YES class

[2] https://fasttext.cc

[3] https://www.nist.gov/tac/

Table 4. Evaluation results for event-argument linking. We report the performance of different model architectures.

	Model	Precision		Recall		F1-score	
		YES	NO	YES	NO	YES	NO
1	CNN without position embedding	0.69	0.44	0.96	0.07	0.80	0.12
2	CNN with position embedding	0.71	0.50	0.90	0.20	0.80	0.29
3	**Bi-LSTM + CNN with position embedding**	**0.74**	**0.47**	**0.81**	**0.38**	**0.77**	**0.42**
4	Stacked CNN without position embedding	0.69	0.43	0.94	0.09	0.79	0.15
5	Stacked CNN with position embedding	0.72	0.38	0.66	0.45	0.69	0.41
6	Bi-LSTM + stacked CNN with position embedding	0.73	0.42	0.74	0.41	0.73	0.42

as compared to all the other models except Model 5 and Model 6. However, *F1-score* for NO class is better than all the models and is equal to that of Model 6.

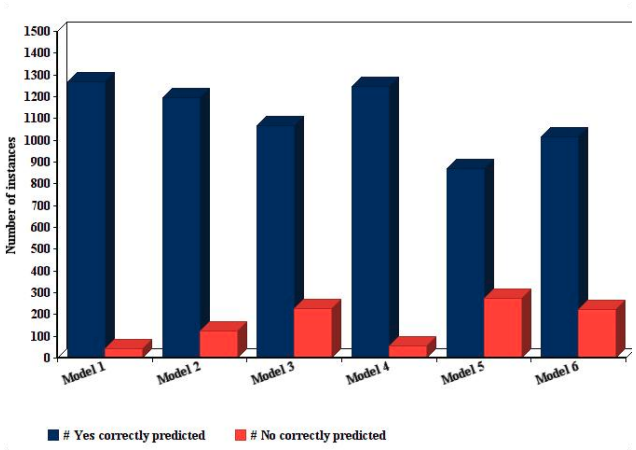

Fig. 3. Plot showing the number of correct predictions for each of the different architectures with respect to both the classes.

4.4 Error Analysis

We carry out an error analysis of the predictions of our proposed model in order to have an appropriate understanding of the system. Our analysis reveals that, out of 1934 instances in the test data, there are 625 instances for which the model has failed to correctly predict the label between the corresponding event and argument. To perform error analysis, we group the instances depending on the position of the event and argument present in the instance, i.e. whether an event lies to the left or to the right of the argument in the instance.

Based on this, we find that instances where event lies to the left and argument to the right, the system fails to detect the link in some cases when the argument

consists of numeric figures(43,727 हेक्टेयर फसल नष्ट (**Translation :** 43,727 hectares of crop destroyed). However, the system predicts the links correctly when the argument consists of a time argument in the form of(2009, 19 अगस्त 2017 (**Translation :** 19 August 2017)).

In the group of instances where event lies to the right and argument to the left, we find that the system fails to correctly predict the link involving a time argument of the type(13 दिसम्बर (**Translation :** 13 December), सुबह 8.20 (**Translation :** morning 8.20)). However, it predicts the link involving other types of argument involving numeric figures like (प्रदेश के कुल 30 जिलों (**Translation :** Total 30 districts of the state)).

This subsection shows a few of the instances where the model has gone wrong in predicting the label between the events and arguments. The word or phrase in red indicates event trigger and the word or phrase in blue indicates argument in the following instance:

1. इस बीच परवान प्रांत के प्रांतीय गवर्नर मोहम्मद असीम ने बताया कि प्रांत के दो जिलोंमें हिमस्खलनों से 16 लोगों की मौत हो गई जबकि आठ अन्य घायल हैं।
   Transliteration : is beech paravaan praant ke praanteey gavarnar mohammad aseem ne bataaya ki praant ke do jilon mein himaskhalanon se 16 logon kee maut ho gaee jabaki aath any ghaayal hain.
   Translation : Meanwhile, provincial governor of the Province Province, Mohammad Asim said that 16 people were killed and eight others were injured in avalanches in two districts of the province.
   Actual label : 1
   Predicted label : 0
   Possible reason : Place argument to the left of the event not detected.

2. 8 घंटे तक चली मुठभेड़ के बाद पाकिस्तानी सुरक्षाक्मियों ने ट्रेनिंग सेंटर पर कब्जा किया।
   Transliteration : 8 ghante tak chalee muthabhed ke baad paakistaanee surakshaakarmiyon ne trening sentar par kabja kiya.
   Translation : After 8 hours of encounter, Pakistani security forces captured the training center.
   Actual label : 1
   Predicted label : 0
   Possible reason : Participant argument not detected by the model.

3. पहला झटका सुबह के 630 बजे दूसरा झटका 645 बजे और तीसरा झटका 648 बजे लगा।
   Transliteration : pahala jhataka subah ke 630 baje doosara jhataka 645 baje aur teesara jhataka 648 baje laga.
   Translation : The first blow came at 630 in the morning, the second blow was 645, and the third shock was 648 hours.
   Actual label : 0
   Predicted label : 1
   Possible reason : Repetition of the same event twice. The model might have thought that first झटका is linked to all the three arguments.

4. भूकंप की तीव्रता रिक्टर स्केल पर 4.4 मैग्नीट्यूड मापी गई।
   Transliteration : bhookamp kee teevrata riktar skel par 4.4 maigneetyood maapee gaee.
   Translation : The magnitude of earthquake measured at 4.4 magnitude on the Richter scale.
   Actual label : 1
   Predicted label : 0
   Possible reason : Intensity argument not detected by the model.

5 Conclusion and Future Works

In this work, we have put forward a deep neural approach for event-argument linking for less-resource language like Hindi. The proposed architecture is a combination of a Bi-LSTM network followed by CNN. As there is no readily available data, we have crawled news data from the different online news sources and annotated for our experiments. The evaluation shows the promising results.

We have performed a detailed analysis of the results, and have also evaluated the effect of position embedding that shows better performance. In future, we would create more annotated data, perform event-argument linking throughout the whole document, induce coreference resolution for linking similar events, and incorporating attention mechanism for finding the best argument match for the event.

Acknowledgments. The work reported in this paper is supported by the project titled ”A Platform for Cross-lingual and Multi-lingual Event Monitoring in Indian Languages”, sponsored by IMPRINT-1, Ministry of Human Resource and Development, Government of India. Sovan Kumar Sahoo gratefully acknowledges “Visvesvaraya PhD Scheme for Electronics and IT”, under the Ministry of Electronics and Information Technology, Government of India. Asif Ekbal acknowledges Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia).

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., et al.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015), software available from tensorflow.org
2. Cai, R., Zhang, X., Wang, H.: Bidirectional Recurrent Convolutional Neural Network for Relation Classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics. vol. 1, pp. 756–765 (2016)
3. Chollet, F., et al.: Keras (2015)
4. Feng, J., Huang, M., Zhao, L., Yang, Y., Zhu, X.: Reinforcement Learning for Relation Classification from Noisy Data. In: Proceedings of AAAI. pp. 5779–5786 (2018)

5. Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning Word Vectors for 157 Languages. In: Proceedings of the International Conference on Language Resources and Evaluation. pp. 3483–3487 (2018)
6. He, Z., Chen, W., Li, Z., Zhang, M., Zhang, W., Zhang, M.: SEE: Syntax-aware Entity Embedding for Neural Relation Extraction, (2018)
7. Kim, Y.: Convolutional Neural Networks for Sentence Classification, (2014)
8. Li, B., Zhao, X., Wang, S., Lin, W., Xiao, W.: Relation Classification using Revised Convolutional Neural Networks. In: Systems and Informatics, 4th International Conference on IEEE. pp. 1438–1443 (2017)
9. 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)
10. Ren, F., Zhou, D., Liu, Z., Li, Y., Zhao, R., Liu, Y., Liang, X.: Neural Relation Classification with Text Descriptions. In: Proceedings of the 27th International Conference on Computational Linguistics. pp. 1167–1177 (2018)
11. Santos, C. N. d., Xiang, B., Zhou, B.: Classifying Relations by Ranking with Convolutional Neural Networks, (2015)
12. Schuster, M., Paliwal, K. K.: Bidirectional Recurrent Neural Networks. IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681 (1997)
13. Wang, L., Cao, Z., De Melo, G., Liu, Z.: Relation Classification via Multi-level Attention CNNs. In: Proceedings of the 54th annual meeting of the Association for Computational Linguistics. vol. 1, pp. 1298–1307 (2016)
14. Xu, K., Feng, Y., Huang, S., Zhao, D.: Semantic Relation Classification via Convolutional Neural Networks with Simple Negative Sampling, (2015)
15. Xu, Y., Mou, L., Li, G., Chen, Y., Peng, H., Jin, Z.: Classifying Relations via Long Short Term Memory Networks along Shortest Dependency Paths. In: Proceedings of the 2015 conference on empirical methods in natural language processing. pp. 1785–1794 (2015)
16. Zeng, D., Liu, K., Lai, S., Zhou, G., Zhao, J.: Relation Classification via Convolutional Deep Neural Network. In: Proceedings of the 25th International Conference on Computational Linguistics: Technical Papers. pp. 2335–2344 (2014)
17. Zhang, D., Wang, D.: Relation Classification via Recurrent Neural Network, (2015)
18. Zhang, R., Meng, F., Zhou, Y., Liu, B.: Relation Classification via Recurrent Neural Network with Attention and Tensor Layers. Big Data Mining and Analytics, vol. 1, no. 3, pp. 234–244 (2018)
19. Zhang, X., Chen, F., Huang, R.: A Combination of RNN and CNN for Attention-based Relation Classification. Procedia computer science, vol. 131, pp. 911–917 (2018)