BUPT_PRIS at TAC KBP 2015

Pengda Qin, Chaoyi Ma, Yidong Jia, Wei Wang, Zhengkuan Zhang, Zuyi Bao
Weiran Xu, Jun Guo
School of Information and Communication Engineering
Beijing University of Posts and Telecommunications
qinpengda@bupt.edu.cn

Abstract

In this paper, we describe our system frameworks and detailed processing for TAC KBP 2015 trask. We participate in 3 tasks: Cold start slot filling track, Event Nugget Detection track and Event Argument Extraction track. We mainly utilize end-to-end neural network system to extract required information in above tasks, such as convolutional and recurrent neural network, which have been verified to be effective in Natural Language processing. Meanwhile, pattern matching and some conventional machine learning methods are also employed as auxiliary approach to further enhance system performance.

1 Introduction

The objective of TAC KBP is to develop and evaluate technologies for populating knowledge bases (KBs) from unstructured text. The tasks of KBP 2015 are focus on the aspects of information extraction of entity, relation and event. Our team takes part in Cold Start Slot Filling (SF) task and all subtasks in Event task.

Currently, deep learning technology has been received extensive attention. Meanwhile, the applications of deep neural network in Natural Language Processing obtain outstanding performance (Nguyen and Grishman, 2015; Xu et al., 2015; Nguyen and Grishman 2015). Consequently, we mainly employ end-to-end neural network to accomplish specific information extraction tasks in TAC KBP 2015 track.

In terms of relation extraction, unlike the previous slot filling (SF) task, Cold Start Slot Filling track only need to discover fillers of specific relation type for every query. However, more difficult is that, for some queries, participants need to use the extracted fillers as new query to extract the next-layer fillers recursively. Such rule is a greater test of the accuracy of relation extraction system. In common with conventional slot filling task, we adopt entity expansion to find more candidate sentences, and utilize some text preprocessing methods to remove noise text component and obtain normative sentence expression. Then, according to named entity recognition results, we employ entity indicator (discussed in Section 2,2) method to mark out the position of query and candidate filler in candidate sentence, and use such format of sentence as input for end-to-end relation extraction classifier. With respect of classifier, we adopt Bi-directional RNN (Bi-RNN), which has excellent ability to remember long-distance text sequence memory. In post-processing stage, we combine multi factors to select filler with the highest confidence.

As for event extraction, this is the first time for our team to construct event extraction system. Different from relation extraction, event extraction system doesn't have specific query, and event trigger words are the essential factor for identification of event mentioned sentences. KBP 2015 Event track has two subtasks. Aiming at different subtask requirement, we adopt various approaches. We train a Convolutional Neural Network framework for Event Nugget. For event argument task, we combine 3 methods (pattern matching, MaxEnt based pipeline method and joint model) to accomplish the extraction.

In addition, we also propose some suitable text processing method, which contribute to significant improvement. Specific details will be described in subsequent sections.

The remainder of the paper is organized as follow. Section 2 describes some text processing details for obtaining normative text expression. Section 2-4 respectively present the proposed system description for slot filling, event nugget and event argument tasks. Finally, in Section 5, we conclude the progresses and deficiencies in this competition and point out the part that need to be strengthened in future work.

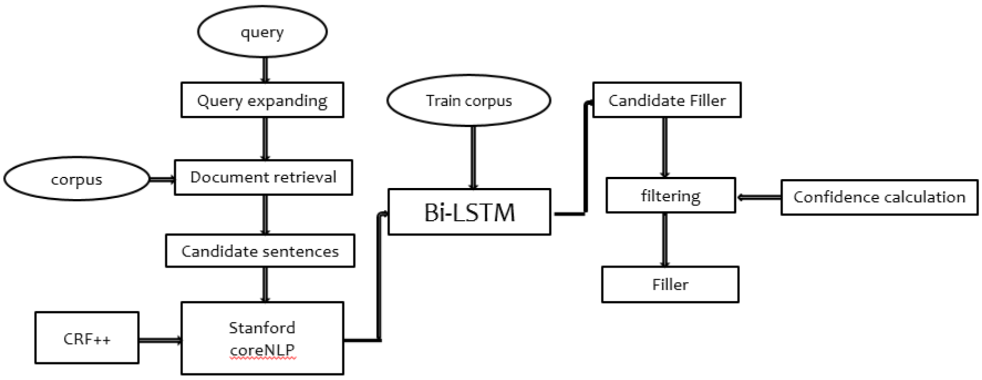

Figure 1: System overview: Cold start slot filling system

2 Cold Start Slot Filling

Figure 1 demonstrates the overview of our cold start slot filling system. This time, we not only reference our former relation extraction system, but also integrate deep learning technology into this extraction system. In general, this system treats relation extraction task as a sentence classification problem. The following subsections describe the details.

2.1 Candidate sentences

Query expansion: The official released query file provides the targeted information to select text region for extracting available fillers. In order to find more candidate sentences, we recursively discover some alternate names of queries. After extracting the first set of sentences based on the original queries format, we leverage high accuracy rule-based method to expand query expression. Then, we use such enlarged query set to extract candidate sentences iteratively. After the 3th time iteration, we find that it scarcely generates new query expression. Additionally, coreference resolution also help us augment candidate sentence set.

Candidate filler: Selecting all non-query words as candidate fillers is time-consuming and it will bring noise components into classifier construction. Consequently, we adopt Named Entity Recognition (NER) to narrow down search range. Current NER tools only support limited types of entity. So, for specific types, such as TITLE, CAUSE, RELIGION, we prepare corresponding list that constructed by words or phrase that extracted from internet resources.

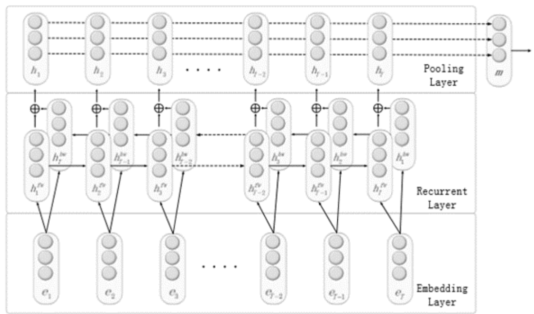

Figure 2: Bi-directional RNN for slot filling

2.2 Bi-RNN classifier

Recurrent neural network (RNN) (Schuster and Paliwal, 1997; Hochreiter et al., 1997) can remember long-distance key information, which is suitable for natural language processing. In this track, we adopt bi-directional RNN which is the combination of forward RNN and backward RNN. The overview of Bi-RNN is presented in Figure 2.

For each step t, hidden vector in one-directional RNN only can store the text sequence information before current state; however, Bi-RNN is capable to integrate the whole sequence information in every hidden vector. Such hidden vectors are calculated by the concatenation of the corresponding output of RNN node in each time step, which are word-level feature. Sentence-level representation is generated by executing max-pooling operation on word-level feature matrix.

In terms of input layer, we leverage the released word embedding set *GoogleNews-vectors-negative300.bin*[1] (Mikolov and Dean, 2013) to initialize every word in input sentence. For out-of-vocabulary words, we randomly initialize the vector representations of them range from [-0.25, 0.25].

[1] https://code.google.com/archive/p/word2vec/

Entity position information: Entity information is the crucial component for relation extraction. In order to indicate the entity position in input sentence, we employ two method: Position feature (Kim, 2014) and Position indicator. Position feature is to give every word relative distance a vector representation, and concatenate it with word embedding as input word representation. Position indicator is a more simple strategy. It uses four position indicators around entity and assigns them vector representations with same dimension as word embedding. The input sentence with query "skin abnormality" and candidate filler "Calluses" can be interpreted as follow:
<e2S> Calluses <e2E> are caused by improperly fitting shoes or by a <e1S> skin abnormality <e1E>

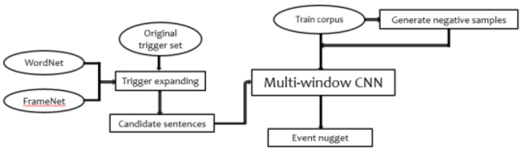

Figure 3: System overview: Event nugget system

Runs	Measure	Precision	Recall	F1
B_P1	CSSF	0.1408	0.1027	0.1188
	CSLDC	0.1741	0.1359	0.1527
B_P2	CSSF	0.1226	0.0378	0.0578
	CSLDC	0.1740	0.0589	0.0880
B_P3	CSSF	0.2232	0.0814	0.1193
	CSLDC	0.2657	0.1094	0.1550
B_P4	CSSF	0.0809	0.0407	0.0542
	CSLDC	0.1232	0.0647	0.0849
B_P5	CSSF	0.0809	0.0407	0.0542
	CSLDC	0.1217	0.0647	0.0845

Table 1: Our cold start slot filling results

2.3 Post-propocess

For some relation types, our system may returns several fillers. However, submitted file only need one submission with highest confidence for one search item. We calculate confidence with two factors: classifier softmax probability and sentence frequency. The accumulation of multiplication of this two value represents confidence score of every filler. We also design various decaying functions to weight the importance of sentence frequency. Such strategy can avoid some particular word or phrase obtaining too high confidence, such as public person.

We also augment the filler expression with POS feature and Chunking result.

2.4 Results

The scores of our submitted results are presented in Table 1.

3 Event Nugget

Event Nugget task is a new member of TAC KBP track. Due to lack of experience, we consulted some classical related papers (Chen et al., 2015), and decided to regard this task as a sentence classification problem. From pre-preparing stage to final output, the system diagram is displayed in Figure 3.

3.1 Trigger word expanding

Currently, the scale of event training corpus has not reach a sufficient level. As a result, the annotated trigger words set in trainset are not enough for extracting adequate candidate event mentioned sentence. In order to overcome this, we employ the linguistic resource WordNet and FrameNet to expand trigger word set. First, words in original trigger word set are transformed to its lemma version. Then, we use the lemma version to discover similar words that are noun, adjective or verb. Finally, we discard the words in stopwords list[2] (supported by NLTK).

Likewise, at the stage of candidate sentence extraction, we use the lemma version of words in corpus to match trigger word set.

3.2 Multi-window CNN classifier

Due to excellent performance of Convolutional neural network in computer vision and speech recognition (Lawrence et al., 1997; Krizhevsky et al., 2012; Abdel-Hamid et al. 2012), the application in NLP area has received extensive attention. In comparison to RNN, CNN has obvious advantages of fast training and easy fitting. Through observation and analysis of trainset, we find the crucial information of event type mainly concentrate on local text snippet rather than word sequence. So we select CNN to identify event type. Because of text expression diversity, CNN with multi window size has better generalization. The overview of our multi-window is shown in Figure 4. Window size is set as [3,4,5,6].

As to input form, it is similar to Cold Start Slot Filling system. We use word embedding to initialize word representations in input layer and indicate the

[2] http://www.nltk.org/book/ch02.html

position of candidate trigger words with the same strategy in Section 2.2.
Because trainset scale doesn't reach a sufficient level, we adopt dropout trick to prevent over-fitting problem (Srivastava et al., 2014).

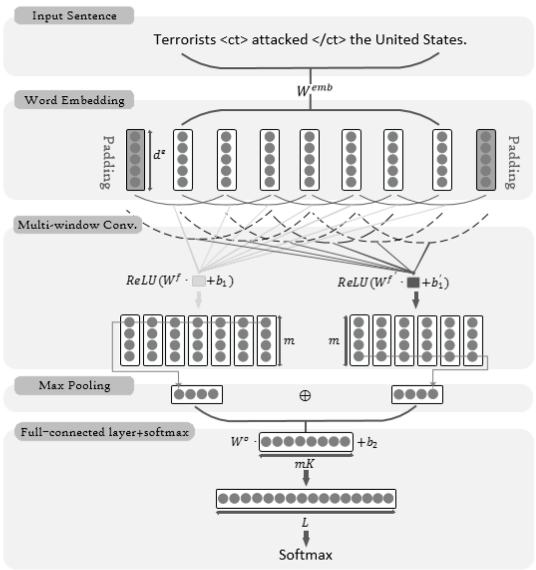

Figure 4: Multi-window CNN for event nugget

3.3 Negative samples

Official corpus only can provide correct instance for training step. However, negative samples are the indispensable ingredient for training classifier. Hence, we propose some solutions:

- Annotate non-trigger word (only select noun, adjective and verb) in correct instance.
- For the sentence in train corpus but not being select as correct instance, if it includes trigger word, such sentence will be put in negative sample set.

3.4 Runs

We submit three runs in 2015 Event Nugget track. The differences between these runs are described as follow:

- **BUPT_PRIS1:** Only annotate words that occurs in trainset trigger word set.
- **BUPT_PRIS2:** Annotate all noun, adjective, verb that appeared in prediction sentence as candidate trigger words.
- **BUPT_PRIS3:** Annotate words that are included in extended trigger word set.

3.5 Results

The score of our submitted results are presented in Table 1.

Run	BUPT_PRIS1			BUPT_PRIS2			BUPT_PRIS3		
	P	R	F1	P	R	F1	P	R	F1
plain	46.60	37.62	41.63	43.51	35.56	39.14	46.73	59.40	52.31
mention_type	35.43	28.70	31.71	27.19	21.44	23.98	32.48	40.43	36.02
realis_status	30.83	24.43	27.26	34.89	27.81	30.95	34.60	43.16	38.41
mention_type+realis_status	23.03	18.27	20.37	21.39	16.41	18.57	23.40	28.42	25.66

Table 2: Our event nugget result

4 Event Argument Extraction

Our event argument extraction system is composed by three feature engineering motheds. The former two methods are conventional information extraction approaches. The third method is a structured model which identify event type and extract event argument simultaneously. Best final results are the integration of predictions by such three models. General framework of our system is presented in Figure 5.

4.1 Pattern matching method

Pattern matching is the traditional and classical information extraction method (Liao et al., 2010; McClosky et al., 2011). Depending on this method, we achieved excellent performance in slot filling task in previous TAC KBP track. Drawing on former experience, we treat trigger word as "query", and regard argument as "filler". Then, we extract short dependency path between them, and select high confidence paths as event pattern. Bootstrapping strategy is adopted to expand pattern set as well. Candidate argument is selected according to identification result of Named Entity Recognition. The accuracy of this method performs well; however, recalling rate is not very satisfactory. So we apply below two approaches to compensate for weakness.

4.2 Pipeline method

This method utilizes a sequence of classifiers to finish argument extraction (Chen and Vincent, 2014):

- **Trigger classifier** is to predict whether a mention is an event trigger.
- **Argument classifier** is to predict, given a trigger and a mention, whether this mention is an argument of an event triggered.
- **Role classifier** is to predict, given an anchor and mention which is an argument of an event triggered, which the argument role to be assigned.

Referring to related papers, we extract a series of features, such as lexical characteristic, n-gram and dependency path.

Because these three classifier are relatively independent, it exists error propagation problem. In order to overcome this, we employ joint model that integrate trigger identification and argument extraction into the same prediction step. The modification increases the interactive relationship between trigger and argument.

4.3 Joint model

Different from above mentioned methods, joint model uses beam search and score function to rank the confidence of (candidate trigger, candidate argument) pair (Qi et al., 2013; Huang et al., 2012; Araki and Mitamura, 2015). Detailed processing can be described as follow:

i. Initialize trigger label set and role label set.

ii. Annotate candidate triggers and arguments in each input sentence.

iii. For each word in sentence, prediction procedure is divided into two parts:
   a) Trigger labelling: Calculate the score of candidate trigger for every event type label, and select K best labels into beam.
   b) Argument labelling: For each trigger label in beam, calculate the score of candidate argument. Then select K best arguments into beam.

iv. Select the instance (trigger, argument) with highest score in beam as final result. In training process, if such final result is not equal to correct label, modify the parameters of score function through gradient descent.

4.4 Runs

BUPT_PRIS1: Only use joint model.
BUPT_PRIS2: Only use pattern matching method.
BUPT_PRIS3: Only use pipeline method.
BUPT_PRIS4: Jointly predict results by these three mentioned methods.

4.5 Results

The score of our submitted results are presented in Table 3.

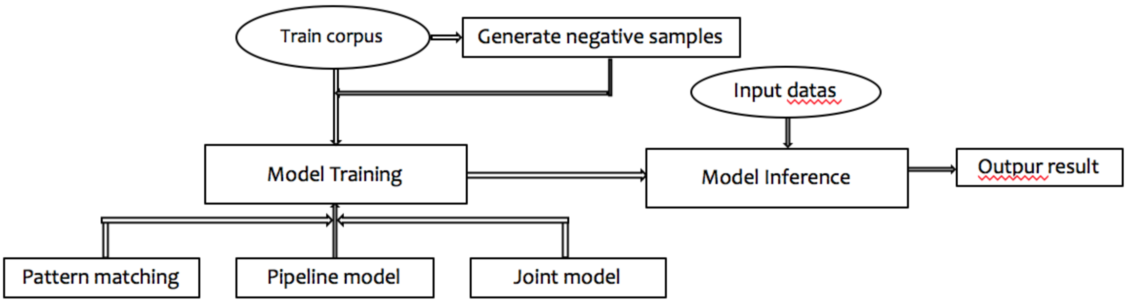

Figure 5: System overview: Event argument extraction system

Runs	precision	Recall	F1	EAArg	EALink	Overall
BUPT_PRIS1	31.79	10.18	15.42	5.81	3.5	4.65
BUPT_PRIS2	32.97	6.03	10.2	3.9	1.39	2.64
BUPT_PRIS3	27.32	10.47	15.14	5.06	3.96	4.51
BUPT_PRIS4	30.65	11.66	16.89	6.3	4.66	5.48

Table 3: Our event argument extraction results

5 Conclusions and future work

This paper describes the detailed methods of our system. In this track, we mainly adopt deep learning to accomplish information extraction. Such approach achieves equivalent or better performance without much effort on costly feature. However, it still exists some problems. For non-canonical texts, such as, forum and web corpus, it performs poorly. This time, we only use text sequence information as input for neural network. In future, we will consider how to integrate linguistic feature into neural network. We are confident that it will improve the performance.

References

Nguyen, Thien Huu, and Ralph Grishman. "Relation extraction: Perspective from convolutional neural networks." *Proceedings of NAACL-HLT*. 2015.

Xu, Yan, et al. "Classifying relations via long short term memory networks along shortest dependency paths." *Proceedings of Conference on Empirical Methods in Natural Language Processing (to appear)*. 2015.

Nguyen, Thien Huu, and Ralph Grishman. "Event detection and domain adaptation with convolutional neural networks." *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Vol. 2. 2015.

Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." *IEEE Transactions on Signal Processing* 45.11 (1997): 2673-2681.

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory."*Neural computation* 9.8 (1997): 1735-1780.

Mikolov, T., and J. Dean. "Distributed representations of words and phrases and their compositionnality." *Advances in neural information processing systems* (2013).

Kim, Yoon. "Convolutional neural networks for sentence classification."*arXiv preprint arXiv:1408.5882*(2014).

Chen, Yubo, et al. "Event extraction via dynamic multi-pooling convolutional neural networks." *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Vol. 1. 2015.

Lawrence, Steve, et al. "Face recognition: A convolutional neural-network approach." *IEEE transactions on neural networks* 8.1 (1997): 98-113.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*. 2012.

Abdel-Hamid, Ossama, et al. "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition." *2012 IEEE international conference on Acoustics, speech and signal processing (ICASSP)*. IEEE, 2012.

Srivastava, Nitish, et al. "Dropout: a simple way to prevent neural networks from overfitting." *Journal of Machine Learning Research* 15.1 (2014): 1929-1958.

Liao, Shasha, and Ralph Grishman. "Filtered ranking for bootstrapping in event extraction." *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, 2010.

McClosky, David, Mihai Surdeanu, and Christopher D. Manning. "Event extraction as dependency parsing." *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, 2011.

Chen, Chen, and Vincent Ng. "SinoCoreferencer: An End-to-End Chinese Event Coreference Resolver." *LREC*. 2014.

Li, Qi, Heng Ji, and Liang Huang. "Joint Event Extraction via Structured Prediction with Global Features." *ACL (1)*. 2013.

Huang, Liang, Suphan Fayong, and Yang Guo. "Structured perceptron with inexact search." *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2012.

Araki, Jun, and Teruko Mitamura. "Joint Event Trigger Identification and Event Coreference Resolution with Structured Perceptron."