The ijk System for EAL at TAC KBP 2016 Event Track

Qifan Yang, Yuzhong Huang, Leiguang Hao, Siliang Tang[*] Fei wu
College of Computer Science, Zhejiang University
{bazinga, hyzme, leiguang, siliang, wufei}@zju.edu.cn

Abstract

In this paper, we describe our system for TAC KBP 2016 Event Argument Extraction and Linking task. Unlike traditional approaches which usually use sequential pipelines, our system utilizes a joint event extraction method predicting event triggers and arguments together. And besides common natural language features (e.g., POS tags), we also make use of promising distributed representations of words and transform these representations into sparse vectors through sparse coding. Experimental results indicate that our joint event extraction model using both common features and sparse vectors performs better.

1 Introduction

Event extraction is important to many machine learning applications, such as information retrieval, machine reading systems or news summarization. Typical event extraction task focus on discovering event triggers with specific types and their arguments. Most state-of-the-art systems extract events and entities in separate stages: trigger identification/classification and argument identification/classification, as a result of which these systems usually fail to correct errors in upstream despite more info is obtained in downstream.

In our system, we adopt a joint event extraction method proposed by (Li et al., 2013). In order to capture the dependencies between triggers and argument as well as explore global features over multiple local predictions, the method formulates event extraction as a structure learning problem and extracts event triggers and arguments simultaneously on sentence level. It uses structured perceptron with early-update to train the joint model and beam search when decoding.

[*]Corresponding author.

Besides commonly used linguistic features such as lemma, synonyms and POS tagging, we also introduce distributed representations of words (Mikolov et al., 2013) which recently have been shown to benefit NLP tasks like parsing (Bansal et al., 2014), named entity recognition (Guo et al., 2014) and sentiment analysis. One major problem what distributed word representations brings is that they are dense and uninterpretable. A method proposed in (Faruqui et al., 2015) transforms any distributed representations of words into sparse vectors through sparse coding, and sparse vectors can then be transformed into binary vectors.

The remainder of the paper is organized as follows. In Section 2, we describe our event argument extraction and linking system in more details. In Section 3, we show some experimental results with our system and our performance in the official TAC KBP 2016 evaluations. Finally, in Section 4, we draw our conclusions on the system.

2 System Description

2.1 Overview

Our system begins with a entity detection module(e.g., Java Extraction Toolkit) that takes raw text as input, and produces named entities within documents in ACE annotation format. These raw documents are also fed into a processing module which runs a suite of tools to extract linguistic features. The processed data is then used as training data for our joint event extraction model, which outputs event mentions and arguments predictions. The output mentions/arguments are also annotated in ACE format, and subsequently fed into a post-processing module which links event arguments into groups and transforms the result into Event Argument Extraction and Linking task output.

2.2 Preprocessing

We begin by extracting named entities and linguistic features from raw documents. In our system, named entity mentions and their coreference links are extracted by running the Java Extraction Toolkit (JET) (Grishman et al., 2005), additionally we also use Stanford Named Entity Recognizer to generate entities and entity annotation outputs from Entity Detection and Linking task. Then we run Stanford CoreNLP toolkit (Manning et al., 2014) on raw text to get sentence tokenization, lemmatization, part-of-speech tagging, and dependency parsing annotations, which are used downstream as features.

2.3 Sparse Coding

Besides features such as lemma and POS tagging mentioned previously, we also introduce distributed representations of words (Mikolov et al., 2013). Sparse coding method proposed by(Faruqui et al., 2015) transforms distributed word representations into sparse vectors, which can then be transformed into binary vectors. And we take these binary vectors as sparse features for the joint event extraction model. First, using word2vec code[1], we train word vectors upon ACE 2005, New York Times news data, and past TAC KBP Event track training corpus. Then these word vectors are fed into sparse coding model[2] to generate corresponding sparse vectors.

2.4 Joint Event Extraction

Given the preprocessed data, we train a joint event extraction model which predicts event triggers and arguments within sentences simultaneously, and code can be found on github[3]. Trigger types and argument roles are based on RichERE ontology.

Raw training documents are split into sentences, and then tokenized into tokens. To train the joint structured perceptron model, for each token we use a variety of features including:

- the current token/lemma
- bigrams of the current token/lemma with words/lemmas within a fixed context window
- part-of-speech tag for the current token
- dependent/governor information from dependency parsing
- nomlex base from noun and WordNet synonyms
- brown cluster for the current token
- possible or nearest entity information for the current token
- sparse binary vector of current token

There are also global features extracted as described by(Li et al., 2013). The structured perceptron classifier takes each sentence as a training instance, and first enumerates each token within the sentence as trigger word, then takes extracted entities within the sentence as event argument candidates and assigns all possible argument roles to each candidate. Such assignments are called configuration, and features for each configuration are represented by extracted token features of sentence and argument role assignment relation. Configuration score is computed according to(Li et al., 2013), and configuration with higher score is better.

Some simple rules are also used to assist the joint event extraction procedure based on observations upon corpus of event task. Take *Attack* event as an example, common trigger words of this kind of events are *attack, bombing, fight, invasion, war, incursion* and so on.

3 Evaluations

We use ACE 2005 and DEFT Rich ERE English Training Annotation V2 as our training corpus. For evaluation purpose, ACE 2005 corpus are split into train/development set as described in (Li et al., 2013), and DEFT Rich ERE English Training Annotation V2 as a whole training corpus. Both model trained on these two corpus are evaluated on TAC KBP 2016 English Event Argument Linking Pilot Gold Standard corpus.

Following past work on event extraction, we report results on micro-averaged precision, recall, and F1 measurements. In order to show the effectiveness of sparse coding vectors of words, we evaluate two forms of our model, one with sparse vectors used and the other without. Results are provided in Table 1.

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

[2]https://github.com/mfaruqui/sparse-coding

[3]https://github.com/oferbr/BIU-RPI-Event-Extraction-Project

Model	ArgPrecision	ArgRecall	ArgF1
sturctured perceptron	0.313	0.049	0.0847
sturctured perceptron+spase coding	0.189	0.066	0.097

Table 1: Effectiveness of Sparse Coding.

We only train our model upon English news or discussion form documents, so we only report results on English corpus. For the released 30k English documents within all evaluation corpus, we need to train another word vectors including words from these documents and then generate sparse vectors. Consequently, we also rebuild structured perceptron model on these new sparse vectors. Results on these official corpus are provided in Table 2.

System	ArgP	ArgR	ArgF1
ijk3	0.052	0.087	0.030

Table 2: Offical results.

4 Conclusions

In this paper, we summarized our system for TAC KBP 2016 Event Extraction and Linking task. Our system utilizes a joint event extraction method based on structured perceptron predicting event triggers and arguments together. The structured perceptron takes both classic linguistic features like lemma or POS tagging and distributed representation of words transformed into sparse vectors through sparse coding. Our experimental results show that distributed representation of words benefit our event extraction task.

Acknowledgments

This work was supported in part by the China Knowledge Centre for Engineering Sciences and Technology (CKCEST), the NSFC (No. 61402401), and the Zhejiang Provincial NSFC (No. LQ14F010004).

References

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. *In Proc. of ACL.*

Manaal Faruqui, Yulia Tsvetkov, Dani Yogatama, Chris Dyer, and Noah Smit. 2015. Sparse overcomplete word vector representations. *In Proc. of ACL.*

Ralph Grishman, David Westbrook, and Adam Meyers. 2005. Nyus english ace 2005 system description. *In Proc. ACE 2005 Evaluation Workshop.*

Jiang Guo, Wanxiang Che, Haifeng Wang, and Ting Liu. 2014. Revisiting embedding features for simple semi-supervised learning. *In Proc. of EMNLP.*

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event ex- traction via structured prediction with global features. *In Proc. of ACL.*

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The stanford corenlp natural language pro- cessing toolkit. *In Proc. of ACL.*

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector spac. *In Proc. of ICLR.*