The Y_dcd_zju Slot Filling System for TAC KBP 2017

Siliang Tang[1], Yujin Yuan[1], Ruowei Jiang[2],
Jinjian Zhang[1], Fei Wu[1], and Yueting Zhuang[1]
1. College of Computer Science, Zhejiang University, Hangzhou, Zhejiang, P. R. China
2. University of Toronto, Toronto, Ontario, Canada
{siliang, yujin}@zju.edu.cn, irenejiang@cs.toronto.edu,
{jinjianzhang, wufei, yzhuang}@zju.edu.cn

Abstract

This report describes the Y_dcd_zju Slot Filling System for the TAC KBP 2017 Cold Start Slot Filling task. To accomplish this task, our system uses text-CNN (Convolutional Neural Network) model to detect sentences that contain pre-defined relations, then we use the NER tag to identify slot fillers. To order to increase the recall of our system, a SVM classifier and a manual trigger words matching algorithm are also used to get all possible slot fillers. Distant Supervision is used to generate training data of our CNN model from massive unlabelled text.

1 Introduction

In this paper, we describe the Y_dcd_zju system for TAC KBP 2017 Cold Start Slot Filling (SF) task, which is organized by NIST. We use Distant supervision (Mintz et al., 2009) to generate training data and trained text-CNN (Kim, 2014) based sentence classifiers for relation extraction. In addition to the neural network, we also use some annotated data to training a SVM classifier, and combine their results when we extract the candidate sentences from data corpus. An overview of our slot filling system is presented in Figure 1.

The reset of the paper is organized as follows: First, an overview of our team's slot filling system (Section 2). Then, we will give some technical details of our system. Finally, we will report the performance of our system in the shared task.

2 System Overview

Slot filling task aim to extract information from unstructured text about entities such as person, organization or some non-entity string like website address. To achieve this, we must solve many technical issues including identify the alias of entities, retrieval side information, coreference resolution, query expansion, training data generation, relation classification and slot filler inference. Therefore, we divide our system into three steps:
1. Data preprocessing
2. Sentence Classifing
3. Slot extracting

3 Data preprocessing

3.1 Preprocessing Documents

In data processing, we need to tokenize the data, split the unstructured documents into sentences, and then get the NER tag of each word. In our system, we use Stanford CoreNLP (Manning et al., 2014) to do this job.

3.2 Expanding Queries

We expect to extract information about the entity given by the queries. However, entities may have many surface names, therefore we need expand queries to address this problem by adding alias for each entity. These aliases were extracted from a Freebase dataset (www.freebase.com).

3.3 Searching Candidates

Sentences which contain the queried entity are regarded as candidates. We use Elastic Search to search the candidate. Elasticsearch is a search engine based on Lucene. It provides a distributed, multitenant-capable full-text search engine with an HTTP web interface and schemafree JSON documents.

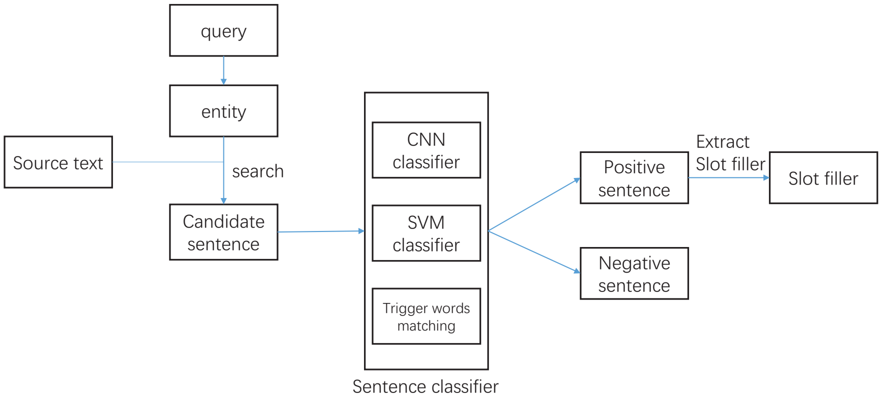

Figure 1: Overview of Slot Filling System

4 Relation Classification

After we collect all the candidates, we need to identify whether the candidates represent required relation, therefore we trained some binary text classifiers to label these candidates.

4.1 Word Embedding

Word embeddings are distributed representations of words that map each word in a text to a k-dimensional realvalued vector. They have recently been shown to capture both semantic and syntactic information about words very well, setting performance records in several word similarity tasks. Using word embeddings that have been trained a priori has become common practice for. enhancing many other NLP tasks. A common method of training a neural network is to randomly initialize all parameters and then optimize them using an optimization algorithm. Recent research (Erhan et al., 2010) has shown that neural networks can converge to better local minima when they are initialized with word embeddings. Word embeddings are typically learned in an entirely unsupervised manner by exploiting the cooccurrence structure of words in unlabeled text. Researchers have proposed several methods of training word embeddings. In our system, we use the word2vec pre-trained Google News corpus (3 billion running words) word vector model (3 million 300-dimension) English word vectors.

4.2 Text CNN Classifier

When we get the embedded sentence by using word vector, our systems applied our Convolutional Neural Networks model to classifier the sentence. Convolutional neural networks (CNN) utilize layers with convolving filters that are applied to local features (LeCun et al., 1998). Originally invented for computer vision, CNN models have subsequently been shown to be effective for NLP and have achieved excellent results in semantic parsing (Yih et al., 2014). For each relation, we need a binary classifier to test whether a candidate represent the relation. This CNN is similar with Kim's way (Kim, 2014). A candidate will be labeled positive when the classifier identify the sentence representing the relation.

4.3 SVM Classifier

Support vector machines are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. Given a set of training examples, each marked as belonging to one or the other of two categories, an SVM training algorithm builds a model that assigns new examples to one category or the other. In addition to the CNN model, we use a SVM classifier to expand positive candidate. We set a threshold to make sure that a sentence is labeled positive only the

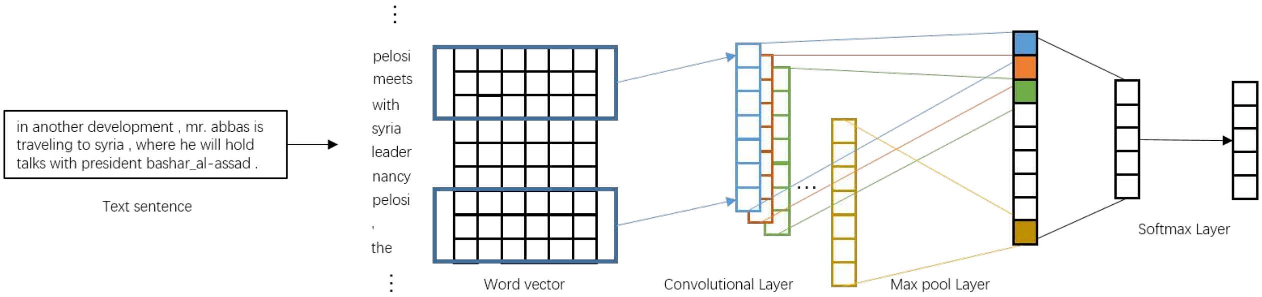
in another development , mr. abbas is traveling to syria , where he will hold talks with president bashar_al-assad .
Text sentence
pelosi
meets
with
syria
leader
nancy
pelosi
,
the
Word vector
Convolutional Layer
Max pool Layer
Softmax Layer

Figure 2: Structure of Text CNN

SVM classifier return a very high confidence score.

4.4 Trigger words matching

Some of the relations in the Slot filling task have obvious trigger words, like 'died in' for relation per:place-of-death. We use a trigger words list to search the candidate sentence, any sentence that match the trigger words will be labeled as positive.

5 Slot extracting

The final step of our Slot filling system is extracting the slot filler from the positive candidates. All of the candidates that be labeled as positive are seem as a representation of the relation. The first entity that match the NER tag we need will be extracted as slot filler.

6 Slot filling result

6.1 Additional Data

As training data, we use the data from LDC. All of the result about evaluation queries are come from LDC2017E25. Additionally, we use the Freebase dataset(www.freebase.com) to get the aliases of entity and slot filler.

6.2 Submission

We have submitted three submissions for the TAC 2017 KBP Cold Start slot-filling track.

- Y_dcd_zju_SF_ENG_1 This submission uses a combined system of a SVM and a CNN model for relation extraction and a more precise pattern-based filler extraction system.
- Y_dcd_zju_SF_ENG_2 This run uses a text-CNN model and a SVM classifier, and just combine their result.
- Y_dcd_zju_SF_ENG_3 This submission only uses text CNN model, which was trained on LDC2015E45 , on the relation extraction part.

	Precision	Recall	F1
Y_dcd_zju_1	0.0902	0.0889	0.0788
Y_dcd_zju_2	0.0853	0.1214	0.0785
Y_dcd_zju_3	0.0549	0.0587	0.0455

Table 1: Results

7 Conclusion

In this paper, we presented an overview of the Y_dcd_zju system for the KBP 2017 English Cold Start Slot Filling (SF) task. This system is an improved version of our last year system. The system mainly applied a combination of distant supervision and Convolutional Neural Networks, and use a SVM classifier and trigger word list to improve the performance. In the future work, we would like to use a more accurate network like PCNN (Zeng et al., 2015), and apply some training strategy like sentence-level attention (Lin et al., 2016) when we use Distant Supervision to get training data.

8 Acknowledgments

This work was supported in part by Chinese Knowledge Center of Engineering Science and Technology (CKCEST), NSFC (No. U1611461, 61402401), 973 program (No. 2015CB352302), Qianjiang Talents Program of Zhejiang Province 2015.

References

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. 2010. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research* 11(Feb):625–660.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. *CoRR* abs/1408.5882. http://arxiv.org/abs/1408.5882.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *ACL (1)*.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*. Association for Computational Linguistics, Stroudsburg, PA, USA, ACL '09, pages 1003–1011. http://dl.acm.org/citation.cfm?id=1690219.1690287.

Scott Wen-tau Yih, Xiaodong He, and Chris Meek. 2014. Semantic parsing for single-relation question answering .

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Conference on Empirical Methods in Natural Language Processing*. pages 1753–1762.