BIT's Slot-Filling Method for TAC-KBP 2013

Sheng Xu[1], Chunxia Zhang[1], Zhendong Niu[2],
Rongyue Mei[2], Junpeng Chen[1], Junjiang Zhang[1], Hongping Fu[2]
[1]School of Software, Beijing Institute of Technology, Beijing, China
[2]School of Computer Science, Beijing Institute of Technology, Beijing, China
{xusheng,cxzhang,zniu,meirongyue,2120131116,
2220120423,fhongping}@bit.edu.cn

Abstract

This paper presents the design and implementation of our English slot filling system. The objective of the slot filling task is to extract attribute values of the given entities. We developed a slot filling system which employs a combinative technique of dependency patterns matching and SVM-based supervision approach. Evaluation results show the strength and weakness of our technique.

1 Introduction

This is the first year that BIT (Beijing Institute of Technology) participated in TAC's Knowledge Base Population Track. We participated in two main tasks of TAC 2013 knowledge base population track: English Slot Filling and Slot Filler Validation. English Slot filling is the task of extracting attribute values of a given entity from large collection of documents. Our approach is based on dependency patterns matching and SVM-based supervision method, which is similar to the works of Li et al. [2011], Grishman et al. [2010] and Sun et al. [2011]. Li et al. used a rule-based method, conditional random field (CRF) and maximum entropy (ME) classification methods to solve the slot filling task. Grishman et al. applied the word sequence patterns and dependency patterns to extract slot values and used a bootstrapping method to build patterns. In our work, a large number of dependency patterns are manually constructed, and synonyms of trigger words of patterns are used to extend those patterns. Moreover, dependency patterns and the support vector machine classifier are employed to extract attribute values of entities.

The rest of the paper is organized as follows. Section 2 presents the slot filling system architecture. We give the main components in sections 3, 4, 5 and 6. In Section 7 we present our experimental results. The conclusions are drawn in Section 8.

2 System Architecture

Our slot filling system consists of three main modules: passage retrieval and preprocessing, query expansion and slot-value selection. We first use two approaches based on dependency patterns matching and SVM-based supervision method to extract attribute values of a given entity. Then, we integrate the results of two approaches to output the ranked attribute values of a given entity. Fig.1 shows the framework of our English slot filling system.

3 Passage retrieval and preprocessing

We have implemented a two-stage passage retrieval module [Byrne 2011]. In the first stage, we retrieve a set of documents in response to the target entity in the initial query using Lucene package. For the entities of ORG type, if the query is an abbreviation of a certain organization, the full name which is obtained through query expansion is also used as a query for documents retrieval. First, we extract the top 200 documents in the search

results of Lucene. Then the documents whose weights are less than a threshold are removed. The threshold is set as 0.4 in our experiment. If there are less than 50 documents left, then the top 50 documents are chosen.

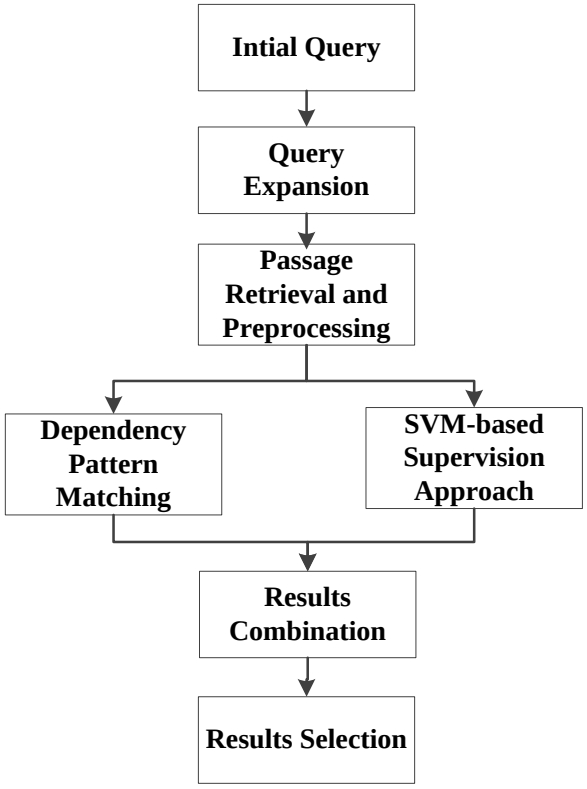

Fig 1. The Framework of Our Slot Filling System

In the second stage, we focus on passages retrieval. We extract all the passages that contain the target entity (including initial and expanded queries) from the retrieved documents. Stanford CoreNLP is used for tokenization, lemma, part of speech tagging, named entity recognition, dependency parsing. Meanwhile, many proper nouns which are collected through wikipedia and Google are utilized for NER.

For some particular slots (including org: website, per:alternate_names and org: alternate_names), a rule-based method is employed to extract their slot values. For the slot org: website, we acquire the websites which contain the target entity from all the related documents. For the other two slots, the extended queries which coexist with the initial query in a document are regarded as slot values.

For other slots except the three slots above, dependency patterns and SVM-based supervision approach are combined to extract slot values. To the slot of location, geo-name lists are employed to identify names of cities, countries and states. In addition, the testing sentences which contain noise words (e.g. say, talk) between entity and candidate slot value are often worthless. For example, the sentence "Bill Gates said to Apple's CEO Steve Jobs…" cannot represent the relation of "Bill Gates" and "Apple". A set of noise words is used to filter this kind of sentences.

4 Query expansion

We use four methods to expand the queries, as follows [Jian 2011]:

(1) If the query has Wikipedia redirect pages, the titles of the redirected pages are used as the query expansion of this entity.

(2) To the entities of PER (person) type, different variations of a person's name are used as the query expansion. For example, given an initial query "Ali Akbar Khan", we can obtain its variants "Ali Akbar", "Ali A. Khan", etc.

(3) To the entities of ORG (organization) type, different variations of an organization's name are used as the query expansion. For instance, given an initial query "IBM", we can obtain the variants "IBM corp. ", "IBM Corporation", "IBM Ltd", etc.

(4) To the entities of ORG (organization) type, if the query is an abbreviation or full name of an entity, we search its full name or abbreviation respectively in the supporting documents. For example, given a query "IBM", we search its full name "International Business Machines" in the supporting documents.

5 Dependency pattern match

In our work, dependency patterns are utilized to extract slot values of given slots of entities. A dependency pattern consists of alternating words and dependency relation labels. For example, we can get the dependency pattern "<Entity> nsubjpass-1 born prep_in <Slot-Value>" from the sentence "Bill was born in 1955" by Stanford dependency parser, where nsubjpass-1 represents an inverse arc (from dependent to head) labeled nsubjpass (passive subject)) [Grishman 2010]. For our slot-value extraction method based on dependency patterns, a lot of dependency patterns are manually collected. Synonyms of slot trigger

words are used to expand the dependency patterns. As an illustration, to the dependency pattern "<Entity>dobj-1 founded prep_in<Slot-Value>", if we find that one of the synonyms of trigger word "found" is "establish", we can expand the initial dependency pattern as "<Entity>dobj-1 established prep_in<Slot-Value>". The number of initial and expanded dependency patterns is about 20,000.

6 SVM-based supervision approach

Support Vector Machine classifier is applied to determine whether a candidate slot value is true or false. For each slot, a binary probabilistic Support Vector Machine classifier (LIBSVM) is trained on the collected dataset. Stanford CoreNLP is used to preprocess the test sentences. A word or a phrase in the sentence whose NER type conforms to the value type of a slot is regarded as a candidate slot value. Only the candidate slot value whose probability is more than a threshold is extracted. The threshold is set as different values in various submitted runs. The training data consists of two parts: first part is the official KBP training data, the second part is the data that are collected through Google. The training data is divided into two parts according to the entity type (PER or ORG). Lexical and syntactic features are used as features for the classifier. It is pointed out that trigger words and noise words as classification features. We regard the common words that are inclined to disturb the result of classification as noise words. For example, for the type of slot per:spouse, the word "like" is often between two persons, but it cannot represent the relation between the two persons.

In the end, the candidate slot values which are extracted by dependency patterns matching and SVM-based supervision method are integrated and ranked according to the used methods and frequencies of those slot values. The confidence score of dependency patterns matching is larger than that of SVM-based method. For the single-answer slot, we select the one with the highest confidence score, while for the list-answer slot, we keep the answers whose confidence scores are larger than a threshold after removing extraneous slot values.

7 Experimental results

Five runs were submitted for the English Slot Filling task this year. Table 1 shows the evaluation results. Our run BIT1 only uses dependency pattern matching method while the others combine dependency pattern matching and SVM-based supervision approaches. The difference among our runs BIT2, BIT3, BIT4, and BIT5 is the different thresholds for filtering the classification results in probabilistic SVM-based supervision approach. The thresholds represent the probability of candidate words which are classified as right slot values. Besides, different slots have different thresholds, according to the confidence of a slot. Result shows that our performance is higher than the median team. We think that our results are benefited from the large number of dependency patterns. The weakness of our system is that the recall is low. There are three main reasons for this fact. First, our result of SVM-based supervision approach is not very well. Second, we don't use coreference resolution technique to extract more candidate slot values. Third, the performance of named entity recognition decreased the precision of our experiment results.

Table 1: English Slot Filling Evaluation Results

	P	R	F1
LDC	0.856	0.571	0.685
Top-1 team	0.425	0.332	0.373
Top-9 team	0.157	0.150	0.153
BIT1	0.579	0.221	0.319
BIT2	0.504	0.229	0.315
BIT3	0.476	0.237	0.316
BIT4	0.230	0.252	0.240
BIT5	0.206	0.260	0.230

8 Conclusions

The paper presents our submission to the English Slot Filling task. We developed a slot filling system which employs a combinative technique of dependency patterns matching and SVM-based supervision approach. The evaluation result shows the feasibility of our approach, which is largely benefited from the large number of dependency patterns. In the future, we will decrease the manual efforts of our approach and try to increase the performance of our approach.

Acknowledgments.

This work is supported by the grant from Chinese National Natural Science Foundation (No: 61272361, 61370137).

References

Yan Li, Xiaoning Li, Hanying Huang, Yang Song, Cheng Chang, Liaoming Zhou Jing Xiao, Dian Yu, Weiran Xu, Guang Chen, Jun Guo. 2011. PRIS at TAC2011 KBP Track. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Ralph Grishman and Bonan Min. 2010. New York University KBP 2010 Slot Filling System. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Ang Sun, Ralph Grishman, Wei Xu, Bonan Min. 2011. New York University 2011 System for KBP Slot Filling. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Lorna Byrne and John Dunnion. 2011. UCD IIRG at TAC 2011. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Xu Jian, Zhengzhong Liu, Qin Lu, Yu-Lan Liu, Chenchen Wang. 2011. PolyUCOMP in TAC 2011 Entity Linking and Slot-Filling. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Heng Ji, Ralph Grishman, Hoa Trang Dang. 2011. Overview of the TAC2011 Knowledge Base Population (KBP) Track. . Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Lorna Byrne and John Dunnion. 2012. UCD IIRG at TAC 2012. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Bonan Min, Xiang Li, Ralph Grishman, Ang Sun. 2012. New York University 2012 System for KBP Slot Filling. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Yan Li, Sijia Chen, Zhihua Zhou, Jie Yin, Hao Luo, Liyin Hong, Weiran Xu, Guang Chen, Jun Guo. 2012. PRIS at TAC2012 KBP Track. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

Benjamin Roth, Grzegorz Chrupała, Michael Wiegand, Mittul Singh, Dietrich Klakow. 2012. Generalizing from Freebase and Patterns using Cluster-Based Distant Supervision for KBP Slot-Filling. Proceedings of Text Analysis Conference. Gaithersburg, Maryland, USA.

BIT at TAC 2013 Slot Filler Validation Track

Rongyue Mei[1], Chunxia Zhang[2], Zhendong Niu[1],
Sheng Xu[2], Junjiang Zhang[2], Hongping Fu[1]
[1]School of Computer Science, Beijing Institute of Technology,
[2]School of Software, Beijing Institute of Technology, Beijing, China
{meirongyue,cxzhang,zniu,xusheng,
2220120423,fhongping}@bit.edu.cn

Abstract

This paper presents the design and implementation of our slot filler validation system. The slot filler validation (SFV) track focuses on the refinement of output from English slot filling (SF) systems by either combining information from multiple slot filling systems, or applying more intensive linguistic processing to validate individual candidate slot fillers. We developed a slot filler validation system which employs three RTE (Recognizing Textual Entailment) methods, including ones based on word overlapping, cosine similarity and token edit distance.

1. Introduction

The task is situated in the Knowledge Base Population (KBP) scenario, and aims at validating the outputted knowledge of the systems participating in the KBP slot filling task by using textual entailment techniques. Given a document D and a set of hypotheses S = {H_1, …, H_n}, the KBP slot filler validation task is to determine whether D entails S, that is D entails Hi(i=1,2,…n). In this framework, S is a set of roughly synonymous sentences representing different linguistic realizations of a relation between a target entity and a possible value (i.e., "slot-filler") of an attribute of this entity (i.e. "slot"). The assumption is that an extracted slot filler is correct if and only if the supporting document entails an hypothesis created based on the slot filler. Our system is developed based on the level of lexical entailment. We employed three methods (Word Overlapping, Cosine Similarity, and Token Edit Distance) to validate the results of slot filling systems, which is similar to those work of Pakray et al. [2011].

The rest of the paper is organized as follows. Section 2 describes the slot filler validation data set. Section 3 presents the slot filler validation system architecture. The experiments results are given in section 4. The conclusions are drawn in section 5.

2. Slot Filler Validation Data Set

The 2013 slot filler validation data set is based on the data constructed for the KBP 2009, 2010, 2011, 2012 and 2013 slot filling task. Specially, our system's training data set consists of over 24,808 T-Hs(text and hypothesis) pairs from the combined RTE-7 training and test sets. The test data set was constructed from the results of KBP 2013 slot filling systems.

3. Our System Architecture

Our slot filler validation system includes two main modules: pre-processing module and RTE module. Fig.1 shows the framework of our slot filler validation system.

The input of our system is pairs of text snippets. For example, to the slot "country_of_birth" of the entity "Tahawwur Hussain Rana", the slot filler is "Pakistan". The input of the example is as follows:

T:"AFP_ENG_20100114.0009",
Hs :{ H_1:Pakistan is the country of birth of TahawwurHussainRana.
H_2: Pakistan is the country where TahawwurHussainRana was born.} .

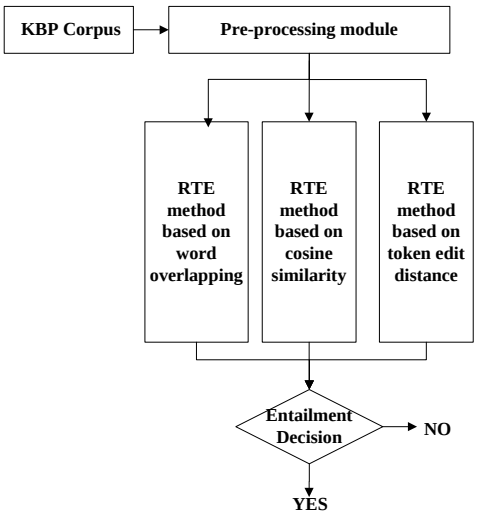

Fig 1: Architecture of Our Slot Filler Validation system

Here, T is the given source document that was cited as supporting the slot filler, and Hs is a set of synonymous hypotheses created from the slot filler by artificial templates. Our system outputs an entailment Boolean value: YES or NO. "YES" means that text (T) entails the set of hypothesis (Hs), and "NO" denotes the opposite meaning.

3.1 Pre-processing Module

In the pre-processing module, first, we use the Stanford CoreNLP tools to perform tokenization, stemming, part-of-speech tagging and full morphological analysis to the source documents. Second, the documents are split into sentences, and are used as the training and test data sets in our experiments. Third, in order to reduce the search space, we employ a filter which is used for all the submitted runs. That filter chooses the sentences which include at least a word within the entity and at least a word within the slot filler. As an illustration, the filtered sentences must contain "Chris" or "Simcox" for "<entity>Chris Simcox<\entity>", and also contain "Tucson" or "Ariz." for the slot filler "<value>Tucson, Ariz.<\value>". If there is no sentence in a document which satisfies the condition above, then this document is tagged as "NO".

Afterwards, a large entailment corpus of T-Hs pairs is created, where each filtered sentence is paired with the corresponding hypothesis set Hs. An example of a T-Hs pair is given as follows, where the value or the slot filler of the attribute "age" of the target entity "Chris Simcox" is "one".

T: "Chris Simcox, one of the movement's co-founders, said three dozen new chapters had formed by mid-November."
Hs: {H1:Chris Simcox is aged one.
H2:ChrisSimcox's age is one.
H3:ChrisSimcox is age one.
H4:Chris Simcox is one years old}.

3.2 RTE methods

In this section, we describe our three RTE methods which based on word overlapping, cosine similarity, and token edit distance.

In the approach based on word overlapping**,** if a word within a hypothesis and one of its synonyms exists in the filtered sentences, then we call the word as "a matched word". WordNet is used to obtain synonyms of words. If n_1 represents the number of matched words and n_2 is the number of words in the hypothesis, we define a matching degree as n_1/n_2. If the degree is not less than 0.9, then the T-Hs pair is considered as an entailment, that is, the T-Hs pair is assigned the value 1. Otherwise, the T-Hs pair is assigned the value -1.

Our RTE method based on cosine similarity is to measure the similarity between two vectors about T-Hs pairs in an inner product space. A vector represents a filtered sentence and another vector denotes one of its corresponding hypotheses. The dimensions of those vectors are the number of different words within the filtered sentence and the hypothesis, while the value of those vectors is the frequency of words in the filtered sentence or the hypothesis.

Hence, the similarity measure between two vectors about T-Hs pairs is a judgment of orientation and not magnitude, that is, two vectors with the same orientation have a cosine similarity 1, and two vectors at an angle of have a similarity of 0. If the value of cosine similarity is 0.5 or more, the text entails the hypothesis, that is, the T-Hs pair is assigned the value 1. Otherwise, the T-Hs pair is assigned the value -1.

In information theory and computer science, the edit distance between two strings of characters usually refers to the Levenshtein distance. Our RTE method based on token edit distance is a token-based version of the Levenshtein distance

algorithm, where edit operations are defined over sequences of tokens of filtered sentences and hypotheses. We compute the value of the token edit distance between a filtered sentence and a hypothesis. If this value is 40 or less, the T-Hs pair is considered as entailment, that is, the T-Hs pair is tagged "YES". Otherwise, the T-Hs pair is tagged "NO". In our experiments, the cut-off values are set based on the experiments carried out on training data set.

4. Experimental Results

For the slot filler validation task, the experimental results of our submitted three runs are shown in Table 1. The result of Run 1 (BIT1_SFV) is that of our RTE method based on word overlapping. The Run2 (BIT2_SFV) and Run3 (BIT3_SFV) are results of our RTE methods based on cosine similarity and token edit distance, respectively. In table 1, "F1" means the average F1 of SFRun (runs of slot filling systems) after applying SFV filter which is proposed by the organizers; "F1′ " is the average F1 of these systems; "Improve" is the average change F1 of SFRun after applying SFV filter.

Table1: Results of Our SFV System

Runs	F1	F1′	Improve
BIT1_SFV	0.0495	0.2079	-0.1583
BIT2_SFV	0.0487	0.2079	-0.1592
BIT3_SFV	0.0384	0.2079	-0.1695

5. Conclusions

This paper presents our submission to the slot filler validation task. We developed a slot filler validation system which employs three methods: word overlapping, cosine similarity and token edit distance. From the Table 1, we can see that the experimental results are not satisfactory. The main reason is that our RTE methods are based on the lexical entailment. Our main challenge is to improve the performance of our system in the future. We will design other RTE approaches which employ syntactic and semantic features and some classification methods.

Acknowledgments.

This work is supported by the grant from Chinese National Natural Science Foundation (No: 61272361, 61370137).

References

Ion Androutsopoulos and Prodromos Malakasiotis. 2010. A Survey of Paraphrasing and Textual Entailment Methods. Journal of Artificial Intelligence Research, 38, 135-187.

Luisa Bentivogli, Peter Clark,Ido Dagan, Danilo Giampiccolo. 2010. The Sixth PASCAL Recognizing Textual Entailment Challenge. Proceedings of the Text Analysis Conference. Gaithersburg, Maryland, USA.

Luisa Bentivogli, Peter Clark,Ido Dagan, DaniloGiampiccolo. 2011. The Seventh PASCAL Recognizing Textual Entailment Challenge. Proceedings of the Text Analysis Conference. Gaithersburg, Maryland, USA.

Asher Stern, EyalShnarch, AmnonLotan, ShacharMirkin, LiliKotlerman, Naomi Zeichner, Jonathan Berant, and Ido Dagan. 2010. Rule Chaining and Approximate Match in Textual Inference. Proceedings of the Text Analysis Conference. Gaithersburg, Maryland, USA.

MilenKouylekov,AlessioBosca,LucaDini. 2011. EDITS 3.0 at RTE-7. Proceedings of the Text Analysis Conference. Gaithersburg, Maryland, USA.

Milen Kouylekov, Yashar Mehdad, Matteo Negri, Elena Cabrio. 2010. FBK Participation in RTE6: Main and KBP Validation Task. Proceedings of the Text Analysis Conference. Gaithersburg, Maryland, USA.

Partha Pakray,SantanuPal, Soujanya Poria,Sivaji Bandyopadhyay, Alexander Gelbukh. 2010. JU_CSE_TAC: Textual Entailment Recognition System at TAC RTE- 6. Proceedings of the 2nd Text Analysis Conference Gaithersburg, Maryland, USA.

Partha Pakray, Santanu Pal,Soujanya Poria,Sivaji Bandyopadhyay, Alexander Gelbukh. 2011. A Textual Entailment System Using Anaphora Resolution. Proceedings of the Text Analysis Conference. Gaithersburg, Maryland, USA.

Ido Dagan, Bernardo Magnini. 2013. Entailment Graphs for Text Exploration. Joint Symposium on Semantic Processing. Trento, Italy.

Matteo Negri, Alessandro Marchetti, Yashar Mehdad, Luisa Bentivogli and Danilo Giampiccolo. 2012. Semeval-2012 Task 8: Cross-lingual Textual Entailment for Content Synchronization. Proceedings of the First Joint Conference on Lexical and Computational Semantics. Montreal, Canada.