PRIS at TAC2010 KBP Track

Sanyuan Gao, Yichao Cai, Si Li, Zongyu Zhang, Jingyi Guan, Yan Li, Hao Zhang, Weiran Xu, Jun Guo
Pattern Recognition and Intelligent System Lab
Beijing University of Posts and Telecommunications
Beijing, P.R. China, 100876
E-mail: sanyuangao@yahoo.com, caiyichaobupt@gmail.com

Abstract

This paper describes our participation in Knowledge Base Population track at TAC2010. In the entity-linking task, we combined machine learning-based methods and rule-based methods to improve the linking results. In the slot filling task, a supervised machine learning method based on CRF model and a rule pattern method were used to select proper answers for slots.

1 Introduction

TAC-KBP track is not an unfamiliar topic for us as we took part in TAC2009-KBP track last year. Just as last year, we participated in both entity linking and slot filling tasks this year. Although having a few differences with TAC2009-KBC track in the requests of TAC2010-KBP track, we just focused on the primary requests and tried to solve them better. So we didn't take surprise slot filling tasks into consideration.

The Entity Linking task is to determine for each query, which knowledge base entity is referred to, or if the entity is not present in the reference KB. And the main difficulties of this task are alias detection (that multiple queries may refer to the same entity using different name variants or different doc ids) and entity disambiguation (that the same query name may refer to multiple entities).

Last year, we thought of entity-linking task as a cross-document co reference resolution problem which employed pronoun resolution technique for summary extraction. Although an ideal method it seems like, its evaluation results were a little unsatisfactory. In TAC2010-KBP track, we consider entity linking task as a retrieval task instead and pay more attention to details. In order to resolve the two difficulties mentioned above effectively, we designed some rules for helping make better decisions. In all, three methods are employed in the entity-linking task, two basic retrieval models and another method we mainly focused on. One of the basic retrieval models corresponds to an optional task of entity linking without wikipedia pages.

The Slot Filling task involves learning a pre-defined set of relationships and attributes for target entities based on the documents in the test collection. We took a two-stage strategy for this task. Firstly, documents related the given query is retrieved by the method using in entity linking task; secondly, the attributes of a given query is determined based on relation extraction technology.

The remaining of this paper is organized as follows. Section 2 and 3 describe systems about entity linking task and slot filling task, respectively. Section 4 presents our evaluation results of the tasks.

2 Entity Linking

We combined machine learning-based methods and rule-based methods for entity linking task, and the rules played a leading role in the system. Fig. 1 shows the framework of EL-RMB system for this task.

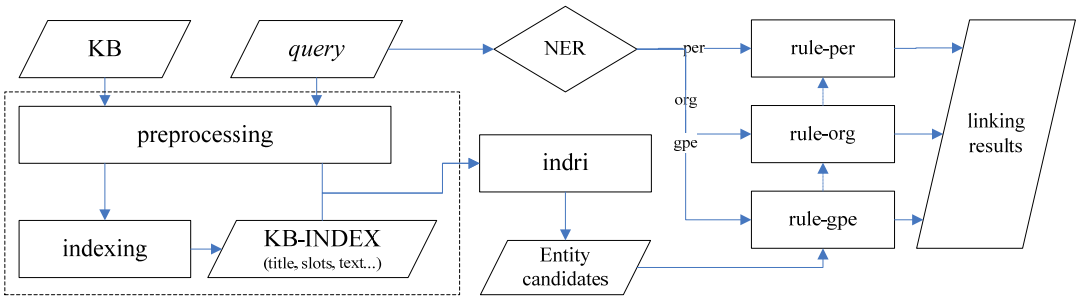

Figure 1: Framework of EL-RMB system

On the whole, EL-RMB system consists of three primary parts: data preprocessing, candidate entities retrieval and linking decision. They are performed sequentially in the system. KB and source corpus are processed first, and then candidate entities are refined by indri [2] which is a language model-based information retrieval tool. Finally, target entity is determined based on a couple of rules.

2.1 data preprocessing

There are mainly two data sets in the track: knowledge base (KB) and source corpus. They are to be processed before the process of entity linking.

1) Acronym expansion for queries in source corpora

   If all the letters in a query are capitals, then the query is considered as an acronym. For any acronym, we try to expand them from the documents associated with the query.

   An N-Gram based approach is used here for acronym expansion. If the acronym has N capital characters, we then check if the initial characters of N continuous tokens in the text are equivalent with the acronym. For example, *Madras Institute of Technology* will be the expanded form of *MIT*.

   We also realized another case of acronym, although it does not happen frequently. Take *CCP* for example, its expanded form may be *Chinese Communist Party* or *Communist Party of China*. In the latter case, the recursive equivalent matching of initials of N continuous tokens is considered for expanding the query.

   If we successfully find the expanded form of a given query, the query will be replaced by its expanded form for querying.

2) Seeking for synonyms for queries

   One of the difficulties of entity linking task is alias detection. To resolve this difficulty, we try our best to find aliases, nicknames or synonyms for a given query as many as possible.

   The *page* table and *redirect* table from wikipedia are used here for achieving this aim. We can get alias, synonyms or abbreviations for some queries by querying the two tables.

3) Generating acronyms and synonyms for titles of nodes in KB

   The same algorithms as finding anonyms and synonyms for queries are used for finding acronyms or synonyms of titles of nodes in KB.

If we successfully find acronyms or synonyms for the title of a node, they will be added with two fields: *ABBRS* and *SYNOS*, which will be used in indexing.

In our point of view, ABBRS, SYNOS have the same meanings with title. So, when we say the title of a node in KB later, it means the combination of all the three fields with *or* relationship between each other. Similarly, a query stands for a combination of the query string and its abbreviations and synonyms.

4) Indexing KB

KB contains thousands of entities. In order to generate entity candidates easily for each query, we index all the KB in advance. And our model is essentially a retrieval model.

We first convert each node in KB into the format of trecweb, and then use indri [2] to build KB index with fields like title, facts, ABBRS, SYNOS and text in each node.

5) Named Entity Recognition (NER)

In order to make rules better, we first divide queries into 4 categories: PER, ORG, GPE and UKN, according to the types of queries. The type a query corresponds to is identified by Stanford Named Entity Recognizer [3].

Stanford NER is also used to recognize the entities in the query-associated documents and the entities in the text of KB nodes. As a matter of convenience, *QE_SET* is used for representing some query-related entities, and *NE_SET* stands for some node-related entities.

2.2 Candidates generation

In order to find out the unique entity in KB for a given query, we first produce a candidate entity list for the query. And it is provided that the KB index has been built in advance.

We employed three methods in entity linking task, and different methods correspond to different strategies. The two basic models correspond to relatively simpler query strategies and simpler decision strategies. In the first basic model, the entity candidates are generated by querying the titles of nodes in the KB index. In the second basic model, the entity candidates are generated by querying the text of nodes in the KB index. In the model we mainly focus on, the entity candidates are generated by querying the combination of title, facts, ABBRS and SYNOS and all the words in the query are an *or* relationship.

If there are more than *N* entities candidates, just top *N* candidates (ordered by indri rank score) are preferred. N is set to be 20 in our experiments.

2.3 linking decision

We must make further decision to select exactly one entity from entity candidates or NIL for a given query. In the two basic models, we just select top one candidate for target entity or NIL for empty candidates set. For the third model, some rules are made for better decisions, which are described as follows.

2.3.1 Common rule

This rule has the highest priority, and it is suitable for queries of any type, which goes like this: if the query matches the title of the entity node equivalently, then the node is considered to be the answer of the query, or the target entity is identified by other rules.

2.3.2 rule-per

This rule is used for queries with *PER* type when no entity candidate meets the common rule.

The rule for person goes as follows:

1) Top N entity candidates are retrieved for the query by Indri, which are considered to be the most related KB nodes with the query.
2) For each pair <query, candidate>, a score is given for its correlation.
   a) *QE_SET* and *NE_SET* are collected.
   b) The number of entities in the intersection of the two sets is counted.
3) If the top one's score is higher than our threshold value, the candidate is considered to be the answer for the query; or the next pair <query, candidate > will be tested. If no candidate meets our score remand, NIL is preferred.

2.3.3 rule-org

This rule is used for queries with *ORG* type when no entity candidate meets the common rule.

The rule for organization goes as follows:

1) Top N entity candidates are retrieved for the query by Indri.
2) Refining entity candidates by selecting the nodes whose title contains all the words in the query.
3) If there is only one refined entity candidate, the candidate is considered to be the answer, or NIL is preferred.

2.3.4 rule-gpe

This rule is used for queries with *GPE* type when no entity candidate meets the common rule.

The rule for Geo-Political goes as follows:

1) Top N entity candidates are retrieved for the query by Indri.
2) For each pair <query, candidate>, the title of candidate entity is separated by comma.
3) If the title of the query occurs as the first part of the candidate title, the candidate is considered to be the answer for the query; or go to next pair <query, candidate> for further testing. If no candidate meets the rule, NIL is a preferred answer.

Take query *China* for example, *China, Asia* is a preferred answer while *Beijing, china* is not.

3 Slot Filling

Fig. 2 shows the framework of the slot filling system, which consists of four primary parts: Entity Retrieval, Entity Type Recognition, Relation Extraction and Resolution Decision.

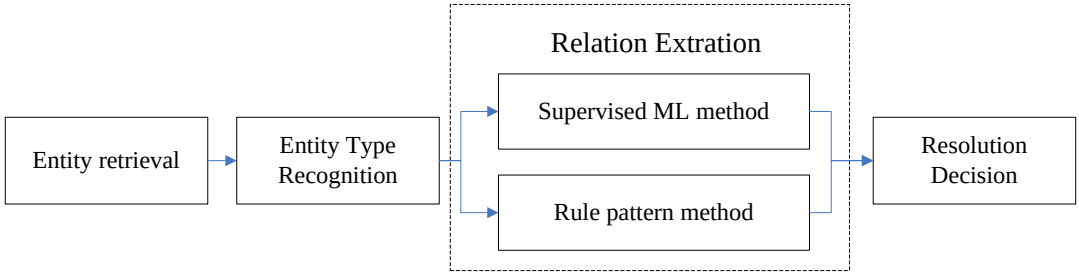

Figure 2: framework of Slot Filling system

Entity Retrieval adopts the same algorithms used in Entity Linking system. The different between them is the source of index. The index was built on the source corpora here.

In the first stage of slot filling, we get top 25 related documents from source corpora for each target entity. And then, Entity Type Recognition is used to recognize entities and time. Person, Organization and Location are recognized by Stanford NER. Time is recognized by regular expression. Relation Extraction and Resolution Decision are shown in detail in the following.

3.1 Relation extraction

Rule pattern-based method and supervised machine learning-based method are used here for relation extraction. In the slot filling system, the rule pattern method has higher priority than the supervised machine learning method. Both methods are described detailly as follows.

3.1.1 Relation extraction based on rule pattern

In the rule pattern method, a set of rule patterns are designed to help filling the slots. Each rule pattern is a regular expression, which mainly composed of four parts: *Target Type* (type of the target entity), *Slot* (the slot name), *Domain Type* (type of the relation answer), and *Keywords* (typical words related to the slot). Tab.1 explains the composition of a rule pattern in detail.

Table 1: composition of a rule pattern

fields	descriptions
Target Type	PER (person) and ORG (organization)
Slot	26 slots for person and 16 slots for organization need filling. Details for the slots can be found in *Task Description for KBP at TAC 2010*.
Domain Type	All the slot values are categorized into 12 domain types, which are shown in Tab. 2. PER, ORG, and LOC are recognized by Stanford NER. DATE, URL and NUM are recognized by regular expressions. Domain types for ORIGIN, DEATH, SCHOOL, TITLE, RELIGION and CHARGE are mainly from lists of candidates which come from the training date in KB
Keywords	Each slot has one or more keywords, which are important for relation extraction. Take the word *born* for example, the sentence contains *born* may contain important information about slots *per:date_of_birth*, *per:country_of_birth*, *per:stateorprovince_of_birth* or *per:city_of_birth*

In the rule pattern method, the *Target Type* and *Domain Type* are recognized firstly, and then slot values were extracted sentence by sentence with the pre-defined rule patterns.

For instance, a rule pattern for slot *per: date_of_birth* may be like this: *"per:date_of_birth\t<PER>([^<]*?)</PER>.*?born.*?<DATE>([^<]*?)</DATE>"*, where *Target Type* is *PER*, *Domain Type* is *DATE*, *Slot* is *per:date_of_birth* and *Keyword* is *born*. $1 in the regular expression stands for the target entity, and $2 stands for the birthday of the person.

Generally, each slot is distributed with one or more rules. And there are 125 rules in all. The number of keywords is almost 100.

3.1.2 Relation extraction based on supervised machine learning method based on CRFs

It is a supervised machine learning method and the system extracts various kinds of features from the context. The relation extraction problem is regarded as a classification problem and a model named CRFs (Conditional Random Fields) is adopted. Features used in the model are listed as the following:

(1) Token pair: the first token is the target entity and the second is the relation token which may be an entity or not.

(2)Word features: the three words before and behind the target entity.

(3)POS features: the POS tags of the three words before and behind the target entity.

(4)Sequence features: the token order between the two tokens.

(5)Verb location features: the position of the verb and the number of words between the verb and the target entity.

(6)Entity location features: the position of the target named entity in the sentence.

(7)Appearance feature: if the token pair appears in the same sub-sentence, the feature is 1.

(8)Number feature: the number of words between the two tokens.

(9)Verb feature: the verb context.

(10)Other entity feature: 1 if another named entity occurs between the token pair else 0

(11)Type feature: the entity type of the token pair, such as PER, if the second token is not a named entity, the feature is set to NULL.

Classifiers based on CRFs were trained with these features. To improve the performance of the classifiers, the training data were divided into two parts according to the entity type (PER or ORG). And each of two parts was further divided into two smaller parts, which is determined by whether the second token of the toke pair is a named entity or not.

After the four classifiers were trained, relation extraction was done for kinds of slots.

3.2 Resolution Decision

It is possible that a slot has more than one different result by using the methods described above. In order to select a better result for a slot, a score function is designed to evaluate the correlation of the slot with the slot result, which is described as follows:

$$Value(Q, slot, doc) = \mu \times Score_{EL}(Q, doc) + (1-\mu) \times Score_{SF}(Q, slot),$$

where Q stands for query, *slot* stands for the value for the slot, *doc* is the document from which the *slot* comes, μ is weighting parameter for balancing the proportion between $Score_{EL}$ and $Score_{SF}$, which is distributed in the interval [0, 1].

Value(Q,slot,doc) is determined by both $Score_{EL}$ and $Score_{SF}$. $Score_{EL}$ means the similarity between query Q and document *doc*, which comes from Entity Retrieval. $Score_{SF}$ is the probability of viewing the current result as the final slot answer. For rule pattern method, the $Score_{SF}$ is set to be 1.

Table 2: Domain Type and Slots

PER		ORG	
Domain Type	Slots	Domain Type	Slots
PER	per:alternate_names per:spouse per:children per:parents per:siblings per:other_family	PER	org:alternate_names org:members org:shareholders org:founded_by org:top_members/employees
ORG	per:member_of per:employee_of	ORG	org:parents org:members org:member_of org:shareholders org:subsidiaries
LOC	per:country_of_birth per:stateorprovince_of_birth per:city_of_birth per:country_of_death per:stateorprovince_of_death per:city_of_death per:countries_of_residence per:stateorprovinces_of_residence per:cities_of_residence per:member_of per:employee_of	LOC	org:member_of org:city_of_headquarters org:country_of_headquarters org:stateorprovince_of_headquarters
DATE	per:date_of_birth per:date_of_death	DATE	org:founded org:dissolved
NUM	per:age	NUM	org:number_of_employees/members
ORIGIN	per:origin	URL	org:website
DEATH	per:cause_of_death	RELIGION	org:political/religious_affiliation
SCHOOL	per:schools_attended		
TITLE	per:title		
RELIGION	per:religion		
CHARGE	per:charges		

4 results of runs

4.1 Entity Linking

Three runs were submitted for the entity linking task, with each corresponding to one module mentioned above. Tab. 3 shows the best evaluation results which comes from our EL-RMB system.

Table 3 : Entity Linking Task Evaluation Results

Our best evaluation results					
2250 queries		**0.7329**			
1020 non-NIL		0.6010			
1230 NIL		0.8423			
Evaluation results for different entity types					
750 ORG	**0.7413**	749 GPE	**0.6662**	751 PER:	**0.7909**
304 non-NIL ORG	0.4901	503 non-NIL GPE	0.6859	213 non-NIL PER	0.5587
446 NIL ORG	0.9126	246 NIL GPE	0.6260	538 NIL PER	0.8829

4.2 Slot Filling

Two runs were submitted for the slot filling task. And the training data for supervised machine learning method is from ACE2008 Evaluation for Relation Detection and Recognition. Tab. 4 shows the better evaluation result when μ was set as 0.5. For list-valued slots, only top 5 answers were selected according to *Value(Q,slot,doc)*.

Table 4 : RESULTS OF THE SLOT FILLING TASK

	Precision	Recall	F1
BuptPris1	0.14043355	0.14410058	0.14224343

5 References

[1] http://nlp.cs.qc.cuny.edu/kbp/2010/
[2] http://www.lemurproject.org/indri/
[3] http://nlp.stanford.edu/software/CRF-NER.shtml