An Introduction to BLCU Personal Attributes Extraction System

Dong YU, Cheng YU, Gongbo TANG, Qin QU, Chunhua LIU, Yue TIAN, Jing YI
College of Information Science, Beijing Language and Cultural University
Beijing 10083, China
yudong_blcu@126.com

Abstract

We describe our methods for share task of personal attributes extraction. We divide all 25 attributes into several categories and propose 4 kinds of pipelines to carry out value extraction. There are two stages in the process. The first stage uses CRF model or regular expression based extractor to produce initial answers. In the second stage, we propose two methods to filter out mistake answers: protagonist dependency relationship based filter and attribute keywords based filter.

1 Introduction

In this paper, we describe the BLCU-PAE system for CIPS-SIGHAN 2014 bakeoffs. The Personal Attributes Extraction (PAE) in Chinese Text Task is designed to extract person specific attributes, like date of birth and death, family relationships, education, title etc. from unstructured Chinese texts. The corresponding techniques play an important role in information extraction, event tracking, entity disambiguation and other related research areas.

In the task, the incomplete attributes of a target person are defined as Slots, i.e. the extracted attribute value need to be filled into these slots. There are 3 kinds of slots, name slots, value slots and string slots, in which only entity name, number/time and string can be filled in. Single-value slots have only one correct answer while list-value slots have a set of answers. There are totally 25 attributes need to be extracted, as shown in Table 1.

Slot filling task has been one of shared tasks in the TAC KBP workshop [Ji and Grishman, 2011] science 2009. In this area, earlier systems generally use one main pipeline that contains 3 stages: document retrieval, answer extraction, and answer combination. Supervised learning normally leads to a reasonably good performance. Both bootstrapping and rule based pattern matching with trigger words are used in [Li, et al., 2013]. Active learning techniques are also used in the task [Chen, et al, 2010]. UNED system introduces a graph structure to solve the problem [Garrido, et al., 2013]. CMUML uses distant supervision and CRF-based structured prediction for producing the final answers [Kisiel, et al., 2013]. Up to now, slot filling remains a very challenging task; most of the shortfall reflects inadequacies in the answer extraction stage.

Type	Attribute
Single slots	city_of _birth, city_of_death, country_of_birth, country_of_death, State_or_province_of_birth, State_or_province_of_death, date_of_birth, date_of_death, cause_of_death, age
List slots	alternative_name, children, cities_of_residence, countries_of_residence , parents, other_family, member_of, siblings, employee_of, spouses, school_attended, religion, charges, titles, state_or_province_of_residence

Table 1: List of all attributes

Our system uses a mixture framework consists of supervised learning and rule based extractor and human knowledge database. We divide 25 attributes into several groups. Each group uses a specific combination of methods for value extraction. Protagonist dependency relationship and key words of attribute are used to filter out suspicious values.

The rest of the paper is organized as follows. Section 2 gives an overview of our system. Section 3 describes models and methods used in the system in detail. Section 4 gives evaluation results and analysis.

Proceedings of the Third CIPS-SIGHAN Joint Conference on Chinese Language Processing, pages 120–125,
Wuhan, China, 20-21 October 2014

2 Overview

At a high level, our PAE system takes a document *d* as input, and produces a set of attributes, each of which contains a specific type *t* and a value *v*. The whole process makes use of a large count of annotated biography corpus collected from BaiduBaike[1] and Chinese Wikipedia[2]. Both supervised machine learning and human designed rules are used for attributes extraction, describes in subsection 2.1.

2.1 The framework

In order to explore various knowledge of person attribute, a large number of biography web pages are collected and divided into sentences. For each attribute, we select a certain number of sentences that contain attribute value, label the position of each value as training data. Meanwhile, attribute value context words are used as keywords for attribute extraction. Figure 1 is the overall framework of our system.

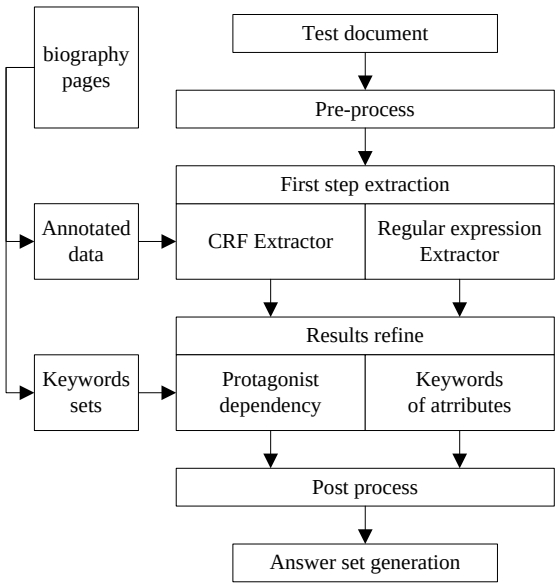

Figure 1: Framework of the system

As shown in Figure 1, the PAE process contains 4 stages:

- Pre-process stage,
- First step extraction,
- Results refine stage,
- Post-process stage.

In the pre-process stage, we divide a test document into sentences, and then carry out a NLP-pipeline on each sentence. Conversely, the post-process stage needs to combine all values extracted from these sentences and produce a final answer set. We will describe both stages in detail in Section 3.

In the first step of extraction, two kinds of extractors are proposed. The first one is CRF extractor. For an attribute, if its context features are obviously difference from others and it has a number of labeled sentences, then attribute extraction can be seen as a sequence labeling problem and CRF model can be used to solve it.

Otherwise, if two or more attributes have similar context, they will have similar features, so CRF cannot distinguish one from another. For example, attributes of *Data of birth* and *Date of death* often appear together in biographies. Data sparse is another obstacle of using CRF, as attribute of "*Religion*" only has dozens of samples. In this situation, regular expression is a better and more direct way for attribute extraction.

Both CRF and regular expression make mistakes during extraction. In our test, there are mainly two kinds of errors:

- Protagonist mismatch,
- Error values caused by models.

So results refine stage is required. In our system, dependency parser is used to filter out values that not related to the protagonist of test document. Keywords of attributes are collected and used to filter out error values. We will describe these methods in detail in section 3.

2.2 Categories of Attributes

The task needs to extract 25 attributes and some of them vary widely from others. Build a model for each attribute can be very consume. So we classify all attributes into several categories, and adopt different extraction pipelines. There are 4 kinds of extraction pipelines in our system. Attribute categories and their extraction pipeline are shown in Table 2.

We train CRF models for attributes related to name entities, such as places, organizations, names. Attributes of *city_of_birth*, *country_of_birth*, and *state_or_province_of_birth* are all place extraction problem, so we train a same CRF model for these attributes. So do place of death and residence.

For attributes that are considered unsuitable for CRF, we use rule based regular expression to extract answers in the first step extraction, including date of birth and death and religion.

For attributes that highly related to person, protagonist dependency between person and values can effectively find out error answers. For

[1] http://www.baike.baidu.com/
[2] http://zh.wikipedia.org

other attributes, for instance *titles*, *member_of*, *cause_of_death*. Other attributes use key words concluded from the training data to refine the answers.

Extraction pipelines	Attribute Categories
CRF only	alternate_names
CRF + protagonist dependency	age, cause_of_death, charges, employee_of, member_of, titles, places of death, places of birth, places of residence
Regular expression only	religion
Regular expression + keywords	date_of_birth, date_of_death, schools_attended, family relationships

Table 2: Attribute Categories

2.3 Resource and toolkits used

We collected more than 40k biographies pages from BaiduBaike and about 6k biographies pages from WikiPedia. The original webpage is very noisy, so we did not use all data for training but select good samples as training data.

We mainly used two toolkits for NLP pipeline, including Chinese word segmentation, POS tagging, NER and dependency parsing: SWJTU Yebol[3] Chinese word segmentation toolkit and LTP-Cloud[4][Che, et al., 2010]. The segmentation accuracy of Yebol can achieve 99.8% and it also used to label time string, place, person name etc. . LTP-Cloud is a cloud based Chinese analysis system that provides dependency parsing, POS tagging and semantic parsing services.

We use CRF++[5] toolkit to train CRF based extractor.

2.4 Data annotation

We annotate start and end of attribute values in sentence level according to the task guideline. Here is an example for *employee_of*: "08 年 7 月 4 日离职【新浪】加入【盛大文学】，任 CEO。" We annotate each category a data set individually. As we used rule-based methods for extraction, such as children, parents, religion, etc, we just summarized their samples and features from training data, and did not annotate them one by one. Finally, we annotate about 25K of positive examples and equal number of negative examples for CRF based extractors.

3 Methods and models

3.1 Pre-process

We adopt a NLP pipeline for each document. Workflow is shown in Figure 2.

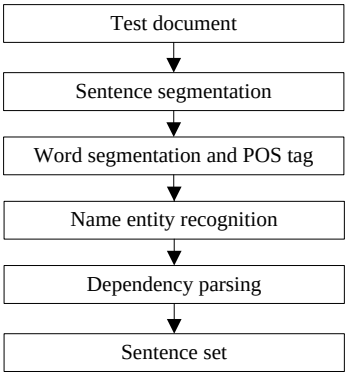

Figure 2: Workflow of pre-process

Pre-process stage is carried out on both train biographies and test documents. We use punctuation to split a document into sentences. Name entity recognition includes time string, person name, place and organization. Dependency parsing is used to find connections between any two words. Pre-process produces a set of sentences all related to document protagonist.

3.2 CRF models training

As mentioned in 2.2, we totally train 10 CRF models. For each model, we use corresponding set of annotated sentences as positive samples, where all values of specific attribute are labeled. Additionally, in order to enhance the model, we also select equal number of negative samples without the attribute. Both positive and negative samples are used for training CRF model.

We use general feature template during training process, mainly include context words and POS tags of context words. The number of training samples for each model is listed in Table 3.

At prediction time, sentences of test document are segmented into word, and tokenized into CRF format, and then the model can tag out all predicted values for the attribute.

[3] http://ics.swjtu.edu.cn/
[4] http://www.ltp-cloud.com/
[5] http://sourceforge.jp/projects/sfnet_crfpp/

Model	Positive Examples	Negative Examples
alternate_names	1230	692
age	513	464
places of birth	10717	1533
places of death	733	1216
places of residence	2194	705
cause_of_death	2122	184
charges	353	939
employee_of	1678	2383
member_of	2330	396
titles	2626	281

Table 3: The statistic of annotations

3.3 Protagonist dependency based filter

CRF based attribute extractor can effectively recognize the existence of attributes in a test sentence and can label out value positions. However, in PAE task, we only need to extract attributes belongs to the protagonist of a test document. For sentences that refers to more than one person, match extracted values with the protagonist can be very difficult. For example, in sentence "他的妹妹 Isobel 因肺炎去世，卡罗瑟斯与妻子 Helen 前往……","肺炎(pneumonia)" is not *Cause_of_death* of protagonist "卡罗瑟斯" but his sister, while CRF always recognize it as a value.

Dependence relationship can help filter out mismatch values. For a test sentence, dependency parsing can convert it into a tree, in which nodes are words. Relationship between any two words can be described by a connected path in the tree. The method is described as follows.

In our test, for each attribute value extracted by CRF or regular expression, we find its head verb and the closest person name in a same sub tree, if the person is protagonist, then we believe that the value is valid. Otherwise, we filter out the value. If test sentence does not have any person or reference, we keep all extracted results by default. Figure X shows an instance of the idea.

Sentence "何雨春，著名画家，1957 年出生于大连。" involves a *title* "画家" and a *place_of_birth* "大连" and a person "何雨春". As shown in Figure 3, two values are dominated by the same verb "出生", the person also in the same sub tree, so both values are available.

On the contrary, in the last instance, the value "肺炎" is dominated by verb "去世", the closest person dominated by the same verb is "Isobel", while protagonist "卡罗瑟斯" is dominated by verb "前往", so the value is filtered out. As shown in Figure 4.

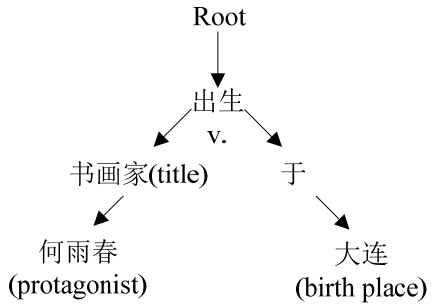

Figure 3: A positive example

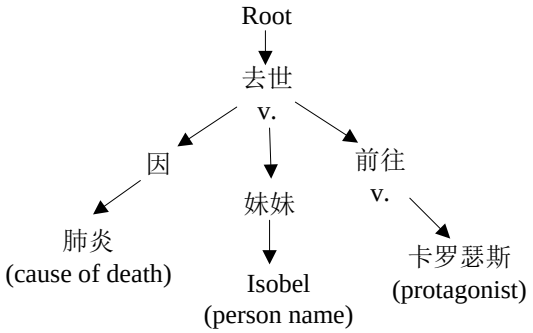

Figure 4: A negative example

In the third instance, "真德秀是南宋后期与魏了翁齐名的理学家。", there are two persons "真德秀" and "魏了翁", and a *title* "理学家". Literally, 魏了翁 is closer to the title than 真德秀, but in dependency tree, 真德秀 and 理学家 are dominated by same verb "是" while 魏了翁 is dominated by verb "齐名", so we think the value "理学家" refers to 真德秀.

3.4 Keywords based filter

Another type of mistakes in our system is caused by defect of models, for example, in "2005.11-2006.1 双流县中和镇人民政府工作，……", the system incorrectly labels "2005.11" as *date_of_birth* in the first step. We find that contexts of this kind of error values are obviously different from right ones. So high frequency context words of attributes can help filter out error values.

The method firstly collects all context words of positive samples of a specific attribute, select a set of words with high frequency as keywords. At test time, we require that there is at least one

keyword in context of extracted value. Otherwise, the extracted value will be abandoned.

Key words based filter can effectively improve accuracy of CRF model. However, it has influence on recall rate. In our system, we collect keywords and used for extracting 5 kinds of familial relationships, schools attended, alternate names, date of death and birth. Table 4 gives some of keywords we used in our system.

Attribute	Keywords
Schools_attended	毕业; 读; 学习; 培训; 肄业; 考 入; 深造; 获得; 学位
siblings	兄; 哥; 姐; 妹; 弟
spouse	妻; 老婆; 媳妇; 爱人; 未婚夫; 老公; 丈夫;
Date_of_death	逝; 牺牲; 卒; 身亡; 去世; 薨; 死; 辞世; 病故; 殁

Tabel 4: Examples of attribute keywords

3.5 Rule and knowledge based methods

Rule based extractor is designed by using regular expression. We use this method in the first step of extraction in *date_of_birth*, *date_of_death*, and *religion*. The first two have very similar contexts so we cannot use CRF to distinguish between them. For the last one, the number of training samples is too small to train a CRF model.

In addition to above methods, human knowledge is also involved in the system, including:

- Country-state/province database,
- Family relationship database,
- Religion database.

As mentioned in 2.2, we train 3 CRF models that can label out birth place, death place and residence place in a test document, regardless level of places. However the PAE task needs to recognize city, state/province and country of places in detail. So we collect a database that contains all countries and most of states/provinces, and divide extracted place sting into different levels, place that is not in database is regarded as city.

Similarly, all family relationships and all religions are also collected. Both databases are used for designing regular expressions and results refine to produce more accurate values.

3.6 Post-process and answer generation

The whole PAE process is done in sentence level and it produces a collect of labeled sentences, one sentence has only one kind of attribute.

In the post-process stage, we need to combine all extracted values together and compute offset of position for each value in original document to generate final XML format answer set. In which all values are written as a record that contain name of protagonist, original document file name, attribute name, attribute values and attribute value offset in the document.

4 Evaluation

4.1 Evaluation matrices

The PAE task takes the same evaluation metrics adopted in the slot filling of TAC KBP. For single attributes, system score is computed by (1), where we set *NumCorrect* to 1.0 when it is zero.

$$Score_{\text{single}} = \frac{NumCorrect}{NumSingleSlot} \quad (1)$$

$$Score_{\text{list}} = \frac{\sum ListSlotValue}{NumListSlots} \quad (2)$$

For list attributes, system score is computed by (2), in which *ListSlotValue* is defined by (3),

$$ListSlotValue = \frac{(F_\beta^2 + 1) * IP * IR}{F_\beta^2 * (IP + IR)} \quad (3)$$

Where F_β = 2 (to weight precision over recall), IP = instance precision and IR = instance recall . Also we set *ListSlotValue* to 0.0, when both *IP* and *IR* are zero. System performance is finally evaluated by (4), that is the average of single attributes evaluation score and list attributes evaluation score.

$$SF_{value} = \frac{1}{2}\left(Score_{\text{single}} + Score_{\text{list}}\right) \quad (4)$$

In the evaluation, both the lenient evaluation and strict evaluation are performed. In the strict evaluation, all instance attributes are compared to the answers while in the lenient evaluation, the offset *string_begin* and *string_end* are ignored.

4.2 Evaluation results

In evaluation, there are totally 90 test persons and 233 test documents. Table 5 shows the evaluation results of our system and the best performance system.

In general, there is still a big gap between our system and the best one. In our system, performances of lenient and strict results are similar. Single score is obviously better than list score, shows that multi-value attributes is more difficult to extract.

Evaluation	Single Score	List Score	SF Value
Lenient (best)	0.6710	0.3438	0.5074
Lenient (ours)	0.4286	0.1888	0.3087
Strict (best)	0.6450	0.3340	0.4895
Strict (ours)	0.4113	0.1739	0.2926

Table5: The evaluation results

4.3 Analysis

Our system still has a lot room for improvements. The first one is to make better use of context in phase level other than sentence level. In our own test, we get more than 0.7 IP score in sentence attributes extraction. However, when it comes to document level, relevance between sentences are more important. In this situation, anaphora resolution and entity link can help to improve the performance of system.

In our system, most of values are extracted based on supervised learning. It is a great challenge for data pre-process and annotation. Bootstrapping style methods can help mining more samples, and active learning framework can be a more effective method to obtain a higher knowledge coverage rate.

Acknowledgements

The research work is partially funded by the Natural Science Foundation of China (No. 61300081, 61170162), and the Fundamental Research Funds for the Central Universities in BLCU (No. 14YJ03005).

Reference

Heng Ji and Raslfph Grishman. 2011. Knowledge Base Population: Successful Approaches and Challenges. *Proc. 49th Annual Meeting Assn. Computational Linguistics.*

Yan Li, Yichang Zhang, Doyu Li, Xin Tong, Jianlong Wang, Naiche Zuo, Ying Wang, Weiran Xu, Guang Chen, Jun Guo. 2013. PRIS at Knowledge Base Population 2013. *Proc. TAC 2013 Workshop.*

Zheng Chen, Suzanne Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew Snover, Javier Artiles, Marissa Passantino and Heng Ji. 2010. CUNYBLENDER TAC-KBP2010 Entity Linking and Slot Filling System Description. *Proc. TAC 2010 Workshop.*

Guillermo Garrido, Anselmo Peñas and Bernardo Cabaleiro. 2013. UNED Slot Filling and Temporal Slot Filling systems at TAC KBP 2013. System description. *Proc. TAC 2013 Workshop.*

Bryan Kisiel, Justin Betteridge, Matt Gardner, Jayant Krishnamurthy, Ndapa Nakashole, Mehdi Samadi, Partha Talukdar, Derry Wijaya, Tom Mitchell. 2013. CMUML System for KBP 2013 Slot Filling. *Proc. TAC 2013 Workshop.*

Wanxiang Che, Zhenghua Li, Ting Liu. LTP: A Chinese Language Technology Platform. *In Proceedings of the Coling 2010:Demonstrations.* 2010, pp13-16, Beijing, China.