A Template Based Hybrid Model for Chinese Personal Name Disambiguation

Zong Hao	Derek F. Wong	Lidia S. Chao
NLP^{2}CT Research Group Department of Computer and Information Science, University of Macau, Macau SAR, China		
MB15463@umac.mo	derekfw@umac.mo	lidiac@umac.mo

Abstract

This paper proposes a template based hybrid model for Chinese Personal Name Disambiguation (CPND). The template makes use of the features of personal role such as discriminating personal name (nickname, stage name), together with the specific context of most frequent words, personal name nearest words named entities, date and time that are effective for this disambiguation task, as well as surrounding context of nominal, verbal and adjectival constituents. The construction of the templates is automatically derived from the articles that maximizes the deviation of different categories of personal names. The extraction algorithm of keyword features based on the distribution of unlabeled data is also proposed in this paper for this challenging task. In addition, an augmented similarity measure for the CPND model has been designed to calculate the similarity between a standard template and an unlabeled text. The final evaluation reveals that the proposed model can achieve the F-measure of 75.75% on the test data.

1 Introduction

The We participated in the CIPS-SIGHAN Joint Conference on Chinese Language Processing and focus on task 2: Chinese Personal Name Disambiguation.

This task is a little different from 2010 SIGHAN task 3[1]. It has given a short description of a certain personal name (here we call this standard classes), and each unlabeled text may belong to three main categories which respectively are a **standard class**, **OUT** class and **OTHER** class.

This task is a little more challenging than 2010 SIGHAN Bake-off task 3, because this task has given us a standard class which usually has less information than an unlabeled text.

This task is very similar to a text clustering problem. Usually most people will use some clustering algorithm, like Xiamen University (Zhu, et al., 2010) and Dalian University (Wang & Huang, 2010) in 2010 SIGHAN Bake-off task3, both of them used Hierarchical Agglomerative Clustering (HAC) algorithm (Jain et al., 1999) to do the clustering. As a conclusion, the most dissimilar in SIGHAN 2010 task3 is that they used different feature set.

For this task, we have a referenced standard class; the clustering for this standard class may not have a good effect. The shortage for this clustering algorithm is that the text must be large enough for this algorithm to extract useful feature, and more importantly the clustering algorithm is very time consuming and highly rely on the feature set. This feature set will add much human effort inside, such as the university name selection, gender selection, job title selection, work experience selection. For this specific task these information may not be enough to distinguish standard classes. Because two standard classes many have some common features. That is the last we want to see. Therefore we design a

[1] http://www.cipsc.org.cn/clp2010/task3_en.html

Proceedings of the Second CIPS-SIGHAN Joint Conference on Chinese Language Processing, pages 121–126, Tianjin, China, 20-21 DEC. 2012

similarity formula to handle the clustering time consuming problem. We pruned most unnecessary calculation. For example, we first calculate the unlabeled text's keyword similarity to each standard class; then further calculate good feature similarity if there is more than one standard has the same keyword similarity. For feature selection, we also design an algorithm to extract the most discriminating words. This original idea of this algorithm is to extract the primitive name or used name. However this personal name information is limited, so we try to use other information as our text feature. Here we proposed a word distribution concept. This word distribution concept refers to the distribution in the whole unlabeled texts. We suppose there is a group of existing words in the standard classes that their sum of distribution is **close to 1**. Since the classification has OTHER and OUT class, we set the expected sum of all distribution is **0.75**. So we suppose the total **OTHER** and **OUT** unlabeled texts are less than 25% of entire texts.

The following sections include Keyword extraction, named entity recognition, model construction, similarity calculation, OUT class solution and other issues. Then we will show the evaluation and conclusion.

2 General Instructions

The keyword extraction in standard class is different from the key word extraction in unlabeled text. Since the words in the standard classes are rare, we have to make full use of these words. In standard classes we extract the personal name (primitive name, used name (name ever used before), stage name, pen name, nickname and so on), organization name (university, company, government organization), other name entity (such as film name, song name, etc.) and other discriminating word as the keyword. In unlabeled text we only extract the named entities as the keywords.

2.1 Keyword Extraction Algorithm

ACL-2010 Keyword extraction is the most significant procedure in our system. We utilize keyword as the most efficient word to associate the unlabeled text with its corresponding standard class.

Many scholars use the bag of words as their keyword, such as AIDA (Yosef et al., 2011), Collective Annotation of Wikipedia Entities in Web Text (Kulkarni et al., 2009), both of them used bag of word strategy. However in this task, we anticipate that there may have some very similar standard classes which are very difficult to distinguish with this bag of words. We suppose that if we use as less keywords as possible, we can distinguish those similar standard classes more easily.

This algorithm is based on this idea, and usually each standard class will have no more than 3 keywords. Using the most discriminating words as our keywords usually gets the best result. We then proposed an algorithm to extract the keywords automatically.

Algorithm 1 Keyword Extraction

Input:
1. $\boldsymbol{SCT}$: Standard classes text;
2. $\boldsymbol{T_{unlabeled\ text}}$: unlabeled text

Output:
1. $\boldsymbol{Kwd}$: Keywords for all standard classes text;

Variables:
$Frag$: segmentation result appending POS;
$Dist$: Distribution of keywords in unlabelled text;

Begin:
For each $SC \in SCT$
  $Frag \leftarrow$ SEG_WITH_POS(SC)
  $Kwd \leftarrow$ CHOOSE_ONE_KWD($Frag$)
  Count[SC] ← 0
  For each $u \in \boldsymbol{T_{unlabeled\ text}}$:
    If $\{\exists w | w \in u, w \in keyword\}$
      $Count[SC] \leftarrow Count[SC] + 1$
      Break
    End if
  End for
End for
$Dist \leftarrow$ SUM($Count$)
While $Dist <$ 0.75 $*$ COUNT ($\boldsymbol{T_{unlabeled\ text}}$)
  $T \leftarrow$ GET_CLASS(MIN($Count$))
  $Frag \leftarrow$ SEG_WITH_POS (T)
  $Kwd \leftarrow$ CHOOSE_ONE_KWD($Frag$)
  For each $\boldsymbol{u} \in \boldsymbol{T_{unlabeled\ text}}$
    If {∃w|w ∈ u, w ∈ keyword}
      Count[T] ← Count[T]+1
      Break
    End If
  End For
  $Dist \leftarrow$ SUM(Count)
End While
Return Kwd
End;

Algorithm 1. Keyword Extraction Algorithm

This algorithm shows the basic strategy of extract keyword. We always follow a rule which is making the distribution keep reasonable. We suppose the distribution should be flat (evenly distributed), hence we got a bad performance on overbalance unlabeled texts.

By this algorithm we can extract really useful keyword. For example, in the test data, after we run this algorithm, we get all keywords as Table 1 below for personal name "白雪 (*Bai Xue*)". Considering the probability of overbalance, not only the keyword but also other useful features together which make the performance of this system much better should be taken into account.

This keyword extraction is only for the standard class, not for the unlabeled texts. It is because that we assume that most of the unlabeled texts have a corresponding standard class, and based on this we design this algorithm, and for the OUT unlabeled texts, we have not figured out a solution.

Standard Class **No.**	**Keyword**
Standard class 1	越剧 (*Shaoxing opera*)
Standard class 2	白百合 (*Bai Baihe*)
Standard class 3	马拉松(*Marathon*)
Standard class 4	配音(*Dub*)
Standard class 5	陈大威 (*Chen Dawei*)
Standard class 6	作家(*Writer*)
Standard class 7	大秦帝国 (*The great Qin empire*)

Table 1: Keywords of Personal Name 白雪 (*Bai Xue*)

2.2 Keyword priority

We set different priority corresponding to different kind of keywords. We consider that the most discriminating words are personal names. When trying to distinguish someone with a same name, other personal titles (such used name, pen name, stage name, etc.) are always the most effective. For example, in standard class 白雪 (*Bai Xue*), 白百合 (*Bai Baihe*) and 陈大威 (*Chen Dawei*) can distinguish these two standard classes efficiently. In Table 2 we list our priority setting for different types of keyword.

Keyword type	**Priority**
Personal name	High
Other named entity	Mid
Other discriminating words	Low

Table 2: Keyword priority

Here all the other discriminating words refer to nouns, and the chosen condition is the distribution of these words in unlabeled text.

3 Named Entity Recognition

Chinese Named Entity Recognition (NER) is more complex than English Named Entity Recognition because it contains a segmentation step before. In this system NER is playing a very important role. For those unlabeled data, it will do NER first. If this system recognizes that the name in this text is not a Named Entity (NE), it will directly assert that this text belongs to the OTHER class. If the name in the text is a NE, we will then mark all the NE in this text to help the later work.

Before we do NER we have to do the Chinese segmentation and Part-of-Speech (POS) tagging. Here this system used ICTCLAS[2] 2011 with additional user dictionary to improve the segmentation and POS tagging accuracy.

Conditional Random Fields (CRFs) is the most popular approach to do NER task. This approach is easy to implement and usually achieve a very high accuracy. Hence this system also used CRFs to do the NER. The CRFs toolkit adopted in this system is CRF++[3] toolkit and used feature is three single characters (before, current, after), three POS tags (before, current, after), some suffix and prefix (s/f) information and three segmentation label sets (before, current, after). The training data set is January-June People's Daily 1998. We get F-measure 91.4% from our test set.

4 Model Construction

We propose a hierarchical personal model for each standard class and unlabeled text. Basically this model consists of four parts:

1. The Keyword, it has the highest priority (K).
2. The second is good features (G), it contains other NE except the keyword, the nearest 10 words and the most frequently used 10 words.
3. The date information word (D).
4. The other information (O) contains all noun, verb and adjective.

In this system, all these features are in different level, we divide features in four levels, and each level's word contributes different weight to the final classification result.

Basically we rule the weight from great to less is K, G, D, O. The keyword in standard class is different from unlabeled text. In the standard class, we use the keyword extraction algorithm,

[2] http://www.ictclas.org/

[3] CRF++: Yet Another Toolkit [CP/OL]. http://crfpp.googlecode.com/svn/trunk/doc/index.html

but in unlabeled text, we check whether this text contains the keyword in standard class is, if it contain, we add this keyword to it K set, otherwise K set will empty.

In this task, the majority of unlabeled texts will have a richer model than standard class, because unlabeled texts have a very high probability of containing a larger size of texts. In some standard class it even contains several single words. Hence, This system also tried to balance the model between the standard class and the unlabeled text. It is defined that if a standard class contains less than 10 words, all this standard class text's words will be added in its model.

5 Similarity Calculation

Most scholars will choose to use cosine similarity between two candidate models as the final similarity between two documents. This method is a measure of similarity between two vectors of an inner product space that measure the cosine of the angle between them. Its value range is from -1 to 1, which is a very good range (no need to do the normalization). Here is an example to explain this method: when calculating the cosine similarity of two candidate documents, firstly convert these two documents into a vector space A and B, the use θ represent the angle between A and B. The similarity then can be calculated using the following equation:

$$similarity = \cos(\theta) = \frac{A*B}{||A||B||} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad (1)$$

Some common vector units are the *tf-idf* words, some user defined useful information (such as university name, job title, age, gender, hometown, etc.).

The biggest advantage of this similarity is its result is already normalized. And the shortage is when converting the model to vector space, and during this procedure some character information will be abandoned. Furthermore this calculation can't solve unsymmetrical length problem (the standard class is usually much shorter than unlabeled text). Therefore we define a formula to overcome its shortage. The formula takes this form:

$$Similarity = \sum_{i=0}^{n} weght * t_i - P \quad (2)$$

t_i Denotes the i^{th} matched word between standard class model and unlabeled model. $Weight$ Denotes a balance factor for different types of words. P Denotes penalty, it depends on the length of the unlabeled text. For each unlabeled text, we will calculate the similarity for each named standard class, and choose the one with largest similarity as its corresponding standard class. When the largest similarity is less than a threshold we will label this text as an OUT class.

6 Out Class Solution

The OUT class enlarged the complexity of this task. The OUT categories are not limited and they are full of uncertainty. Some OUT texts may be very similar to a standard class related text. In this section we defined a formula. It can basically distinguish the OUT class.

To handle the OUT class, we need clustering algorithm. The basic idea is still using the Similarity formula. The detail algorithm is following:

Algorithm 2 OUT Classification

Input:
O : All potential OUT class text
Output:
Label : Label for each OUT class text
Begin
Variables:
$Model$: Consist of a group of features extracted from text;
$Threshold$: A threshold used to determine whether this model is belong to a certain model or not;
For each O ∈ $\{OUT\ text\}$
  If Order$_O = 1$
    $Label \leftarrow OUT\_01$
    $Model \leftarrow$ EXTRACT_FEATURE(O)
  Else
    $T_{model} \leftarrow$ EXTRACT_FEATURE(O)
    $Max \leftarrow -1$
    $Label \leftarrow default$
    For each M ∈ Model
      $Simi \leftarrow$ SIMILARITY$(Tmodel, M)$
      If $Simi < Max$
        $Max \leftarrow Simi$
        $Label \leftarrow M_{label}$
      END If
    End For
      If $Max > Threshold$
        MERGE_MODEL(T_{model}, M)
      Else
        $Label \leftarrow OUT\_$(MAX$(Model_{label}) + 1)$
  Return Label
End For
End;

Algorithm 2. OUT classification Algorithm

7 System Architecture

Our system involves the following steps to do the personal name disambiguation.

For the standard class:

1) Extract keyword and other useful information. Utilize this information to build a model for this standard class.

For the unlabeled text:

1) Do named entity recognition, label all the named entity in this text, if the certain name is not a named entity, marked it as the **OTHER** category.
2) Extract keyword and other useful information (good feature, date information and other nouns, verbs, adjectives).
3) Calculate the similarities against the standard class.

The main architecture of this system is shown in Figure 1.

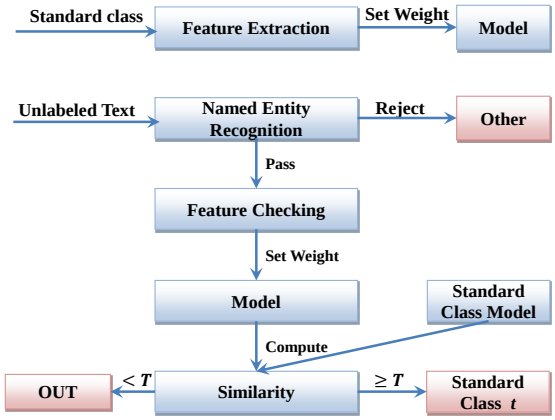

Figure 1: Main workflow. *T* denotes a threshold.

8 Other Issues

There are some other issues about this task, firstly we think the word match method should not be completely matched, we should use a similarity instead. Since our matching approach did not contain large information about the word position. We get a bit lower F-measure after applying a TongYiCiCiLin(同义词词林) based similarity calculation.

We also tried to add the information about the distance to the headword which is the certain personal name by setting weight. Due to the complexity of the unlabeled text, this approach did not show a better result.

9 Evaluation

We followed the formula given by the organizers to calculate the precision rate, recall rate and FB1[4].

We directly list the best test result based on the given so called train set (Table 3):

Personal Name	P	R	FB1
白雪(*Bai Xue*)	0.7447	0.7944	0.7687
白云 (*Bai Xun*)	0.5333	0.7526	0.6243
丛林 (*Cong Lin*)	0.7738	0.8956	0.8303
杜鹃(*Du Juan*)	0.7143	0.9010	0.7969
方正 (*Fang Zheng*)	0.6064	0.9135	0.7289
胡琴 (*Hu Qin*)	0.7577	0.9131	0.8282
华明(*Hua Ming*)	0.8511	0.9770	0.9097
华山 (*Hua Shan*)	0.5062	0.7332	0.5989
Total	0.6859	0.8600	0.7632

Table 3: The evaluation of the training data

And for the competition, our result is in Table 4:

Precision	Recall	FB1
0.7256	0.7923	0.7575

Table 4: The official evaluation of final test.

We only get overall score, not in detail. All these data show that our recall rate is obviously larger than the precision rate. Which means our system is better at detecting the OUT and the OTHER class.

10 Conclusion

We designed an approach for this Chinese Personal Name Disambiguation task. In our approach we firstly removed the OTHER class and then using a name model to distinguish the unlabeled text. We designed a keyword extraction algorithm which is significantly useful in this task. Furthermore, since the recall rate is always larger than the precision rate, our designed formula is also vital.

We implement this system in Python, and our system is highly efficient, in the so called train set, our whole classification procedure cost only 5 seconds, and for the final test set it cost 55 seconds (experiment environment: Inter Core i5 760 CPU and 8GB DDR3 1333 memory).

Acknowledgments

[4] http://www.cipsc.org.cn/clp2012/task2.html

This work was partially supported by the Research Committee of University of Macau under grant UL019B/09-Y3/EEE/LYP01/FST, and also supported by Science and Technology Development Fund of Macau under grant 057/2009/A2.

References

Jain, A. K., Murty, M. N. & Flynn, P. J. 1999. *Data clustering: a review*, ACM computing surveys (CSUR),31(3),264–323.

Kulkarni, S., Singh, A., Ramakrishnan, G. & Chakrabarti, S. 2009. *Collective annotation of Wikipedia entities in web text*, Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 457–466.

Wang, D. & Huang, D. 2010. *DLUT: Chinese Personal Name Disambiguation with Rich*, CIPS-SIGHAN Joint Conference on Chinese Language Processing.

Yosef, M. A., Hoffart, J., Bordino, I., Spaniol, M. & Weikum, G. 2011. *Aida: An online tool for accurate disambiguation of named entities in text and tables*, Proceedings of the VLDB Endowment, 4(12).

Zhu, X., Shi, X., Liu, N., Guo, Y. M. & Chen, Y. 2010. *Chinese Personal Name Disambiguation: Technical Report of Natural Language Processing Lab of Xiamen University*, CIPS-SIGHAN Joint Conference on Chinese Language Processing.

Duan, H. & Zheng, Y. 2011. *A study on features of the CRFs-based Chinese Named Entity Recognition*, International Journal of Advanced Intelligence, 3(2), 287–294.