Coarse-grained Candidate Generation and Fine-grained Re-ranking for Chinese Abbreviation Prediction

Longkai Zhang Houfeng Wang Xu Sun
Key Laboratory of Computational Linguistics (Peking University)
Ministry of Education, China
zhlongk@qq.com, wanghf@pku.edu.cn, xusun@pku.edu.cn

Abstract

Correctly predicting abbreviations given the full forms is important in many natural language processing systems. In this paper we propose a two-stage method to find the corresponding abbreviation given its full form. We first use the contextual information given a large corpus to get abbreviation candidates for each full form and get a coarse-grained ranking through graph random walk. This coarse-grained rank list fixes the search space inside the top-ranked candidates. Then we use a similarity sensitive re-ranking strategy which can utilize the features of the candidates to give a fine-grained re-ranking and select the final result. Our method achieves good results and outperforms the state-of-the-art systems. One advantage of our method is that it only needs weak supervision and can get competitive results with fewer training data. The candidate generation and coarse-grained ranking is totally unsupervised. The re-ranking phase can use a very small amount of training data to get a reasonably good result.

1 Introduction

Abbreviation Prediction is defined as finding the meaningful short subsequence of characters given the original fully expanded form. As an example, "HMM" is the abbreviation for the corresponding full form "Hidden Markov Model". While the existence of abbreviations is a common linguistic phenomenon, it causes many problems like spelling variation (Nenadić et al., 2002). The different writing manners make it difficult to identify the terms conveying the same concept, which will hurt the performance of many applications, such as information retrieval (IR) systems and machine translation (MT) systems.

Previous works mainly treat the Chinese abbreviation generation task as a sequence labeling problem, which gives each character a label to indicate whether the given character in the full form should be kept in the abbreviation or not. These methods show acceptable results. However they rely heavily on the character-based features, which means it needs lots of training data to learn the weights of these context features. The performance is good on some test sets that are similar to the training data, however, when it moves to an unseen context, this method may fail. This is always true in real application contexts like the social media where there are tremendous new abbreviations burst out every day.

A more intuitive way is to find the full-abbreviation pairs directly from a large text corpus. A good source of texts is the news texts. In a news text, the full forms are often mentioned first. Then in the rest of the news its corresponding abbreviation is mentioned as an alternative. The co-occurrence of the full form and the abbreviation makes it easier for us to mine the abbreviation pairs from the large amount of news texts. Therefore, given a long full form, we can generate its abbreviation candidates from the given corpus, instead of doing the character tagging job.

For the abbreviation prediction task, the candidate abbreviation must be a sub-sequence of the given full form. An intuitive way is to select all the sub-sequences in the corpus as the candidates. This will generate large numbers of irrelevant candidates. Instead, we use a contextual graph random walk method, which can utilize the contextual information through the graph, to select a coarse grained list of candidates given the full form. We only select the top-ranked candidates to reduce the search space. On the other hand, the candidate generation process can only use limited contextual information to give a coarse-grained ranked list of candidates. During generation, can-

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1881–1890,
October 25-29, 2014, Doha, Qatar.

didate level features cannot be included. Therefore we propose a similarity sensitive re-ranking method to give a fine-grained ranked list. We then select the final result based on the rank of each candidate.

The contribution of our work is two folds. Firstly we propose an improved method for abbreviation generation. Compared to previous work, our method can perform well with less training data. This is an advantage in the context of social media. Secondly, we build a new abbreviation corpus and make it publicly available for future research on this topic.

The paper is structured as follows. Section 1 gives the introduction. In section 2 we describe the abbreviation task. In section 3 we describe the candidate generation part and in section 4 we describe the re-ranking part. Experiments are described in section 5. We also give a detailed analysis of the results in section 5. In section 6 related works are introduced, and the paper is concluded in the last section.

2 Chinese Abbreviation Prediction System

Chinese Abbreviation Prediction is the task of selecting representative characters from the long full form[1]. Previous works mainly use the sequence labeling strategies, which views the full form as a character sequence and give each character an extra label 'Keep' or 'Skip' to indicate whether the current character should be kept in the abbreviation. An example is shown in Table 1. The sequence labeling method assumes that the character context information is crucial to decide the keep or skip of a character. However, we can give many counterexamples. An example is "北京大学"(Peking University) and "清华大学"(Tsinghua University), whose abbreviations correspond to "北大" and '清华' respectively. Although sharing a similar character context, the third character '大' is kept in the first case and is skipped in the second case.

We believe that a better way is to extract these abbreviation-full pairs from a natural text corpus where the full form and its abbreviation co-exist. Therefore we propose a two stage method. The first stage generates a list of candidates given a large corpus. To reduce the search space, we adopt graph random walk to give a coarse-grained ranking and select the top-ranked ones as the candidates. Then we use a similarity sensitive re-ranking method to decide the final result. Detailed description of the two parts is shown in the following sections.

[1]Details of the difference between English and Chinese abbreviation prediction can be found in Zhang et al. (2012).

Full form	香	港	大	学
Status	Skip	Keep	Keep	Skip
Result		港	大	

Table 1: The abbreviation "港大" of the full form "香港大学" (Hong Kong University)

3 Candidate Generation through Graph Random Walk

3.1 Candidate Generation and Graph Representation

Chinese abbreviations are sub-sequences of the full form. We use a brute force method to select all strings in a given news article that is the sub-sequence of the full form. The brute force method is not time consuming compared to using more complex data structures like trie tree, because in a given news article there are a limited number of sub-strings which meet the sub-sequence criteria for abbreviations. When generating abbreviation candidates for a given full form, we require the full form should appear in the given news article at least once. This is a coarse filter to indicate that the given news article is related to the full form and therefore the candidates generated are potentially meaningful.

The main motivation of the candidate generation stage in our approach is that the full form and its abbreviation tend to share similar context in a given corpus. To be more detailed, given a word context window w, the words that appear in the context window of the full form tend to be similar to those words in the context window of the abbreviations.

We use a bipartite graph $G(V_{word}, V_{context}, E)$ to represent this phenomena. We build bipartite graphs for each full form individually. For a given full form v_{full}, we first extract all its candidate abbreviations V_C. We have two kinds of nodes in the bipartite graph: the word nodes and the context nodes. We construct the word nodes as $V_{word} = V_C \cup \{v_{full}\}$, which is the node set of the full form and all the candidates. We construct the context nodes $V_{context}$ as the words that appear

in a fixed window of V_{word}. To reduce the size of the graph, we make two extra assumptions: 1) We only consider the nouns and verbs in the context and 2) context words should appear in the vocabulary for more than a predefined threshold (i.e. 5 times). Because G is bipartite graph, the edges E only connect word node and context nodes. We use the number of co-occurrence of the candidate and the context word as the weight of each edge and then form the weight matrix W. Details of the bipartite graph construction algorithm are shown in Table 2. An example bipartite graph is shown in figure 1.

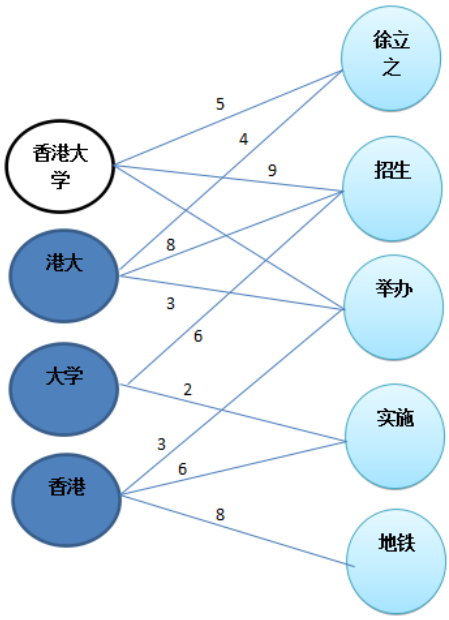

Figure 1: An example of the bipartite graph representation. The full form is "香港大学"(Hong Kong University), which is the first node on the left. The three candidates are "港大", "香港", "大学", which are the nodes on the left. The context words in this example are "徐立之"(Tsui Lap-chee, the headmaster of Hong Kong University), "招生"(Enrollment), "举办"(Hold), "实施"(Enact), "地铁"(Subway), which are the nodes on the right. The edge weight is the co-occurrence of the left word and the right word.

3.2 Coarse-grained Ranking Using Random Walks

We perform Markov Random Walk on the constructed bipartite graph to give a coarse-grained ranked list of all candidates. In random walk, a *walker* starts from the full form source node S (in later steps, v_i) and randomly *walks* to another node v_j with a transition probability p_{ij}. In random walk we assume the *walker* do the *walking* n times and finally stops at a final node. When the walking is done, we can get the probability of each node that the *walker* stops in the end. Because the *destination* of each step is selected based on transition probabilities, the word node that shares more similar contexts are more likely to be the final stop. The random walk method we use is similar to those defined in Norris (1998); Zhu et al. (2003); Sproat et al. (2006); Hassan and Menezes (2013); Li et al. (2013).

The transition probability p_{ij} is calculated using the weights in the weight matrix W and then normalized with respect to the source node v_i with the formula $p_{ij} = \frac{w_{ij}}{\sum_l w_{il}}$. When the graph random walk is done, we get a list of coarse-ranked candidates, each with a confidence score derived from the context information. By performing the graph random walk, we reduce the search space from exponential to the top-ranked ones. Now we only need to select the final result from the candidates, which we will describe in the next section.

4 Candidate Re-ranking

Although the coarse-grained ranked list can serve as a basic reference, it can only use limited information like co-occurrence. We still need a re-ranking process to decide the final result. The reason is that we cannot get any candidate-specific features when the candidate is not fully generated. Features such as the length of a candidate are proved to be useful to rank the candidates by previous work. In this section we describe our second stage for abbreviation generation, which we use a similarity sensitive re-ranking method to find the final result.

4.1 Similarity Sensitive Re-ranking

The basic idea behind our similarity sensitive re-ranking model is that we penalize the mistakes based on the similarity of the candidate and the reference. If the model wrongly selects a less similar candidate as the result, then we will attach a large penalty to this mistake. If the model wrongly chooses a candidate but the candidate is similar to the reference, we slightly penalize this mistake. The similarity between a candidate and the reference is measured through character similarity, which we will describe later.

Input: the full form v_{full}, news corpus U
Output: bipartite graph $G(V_{word}, V_{context}, E)$

Candidate vector $V_c = \emptyset$, $V_{context} = \emptyset$
for each document d in U
  if d contains v_{full}
  add all words w in the window of v_{full} into $V_{context}$
  for each n-gram s in d
  if s is a sub-sequence of v_{full}
  add s into V_c
  add all word w in the window of s into $V_{context}$
  end if
  end for
  end if
end for
$V_{word} = V_c \cup \{v_{full}\}$
for each word v_i in V_{word}
  for each word v_j in $V_{context}$
  calculate edge weight in E based on co-occurrence
  end for
end for
Return $G(V_{word}, V_{context}, E)$

Table 2: Algorithm for constructing bipartite graphs

We first give some notation of the re-ranking phase.

1. $f(x, y)$ is a scoring function for a given combination of x and y, where x is the original full form and y is an abbreviation candidate. For a given full form x_i with K candidates, we assume its corresponding K candidates are y_i^1,y_i^2,...,y_i^K.

2. evaluation function $s(x, y)$ is used to measure the similarity of the candidate to the reference, where x is the original full form and y is one abbreviation candidate. We require that $s(x, y)$ should be in $[0, 1]$ and $s(x, y) = 1$ if and only if y is the reference.

One choice for $s(x, y)$ may be the indicator function. However, indicator function returns zero for all false candidates. In the abbreviation prediction task, some false candidates are much closer to the reference than the rest. Considering this, we use a Longest Common Subsequence(LCS) based criterion to calculate $s(x, y)$. Suppose the length of a candidate is a, the length of the reference is b and the length of their LCS is c, then we can define precision P and recall R as:

$$\begin{aligned} P =& \frac{c}{a}, \\ R =& \frac{c}{b}, \\ F =& \frac{2 * P * R}{P + R} \end{aligned} \tag{1}$$

It is easy to see that F is a suitable $s(x, y)$. Therefore we can use the F-score as the value for $s(x, y)$.

3. $\phi(x, y)$ is a feature function which returns a m dimension feature vector. m is the number of features in the re-ranking.

4. $\vec{w}$ is a weight vector with dimension m. $\vec{w}^T\phi(x, y)$ is the score after re-ranking. The candidate with the highest score will be our final result.

Given these notations, we can now describe our re-ranking algorithm. Suppose we have the training set $X = \{x_1, x_2, ..., x_n\}$. We should find the weight vector $\vec{w}$ that can minimize the loss function:

$$\begin{aligned} Loss(\vec{w}) = \sum_{i=1}^{n} \sum_{j=1}^{k} ((s(x_i, y_i^1) - s(x_i, y_i^j)) \\ * I(\vec{w}^T\phi(x_i, y_i^j) \geq \vec{w}^T\phi(x_i, y_i^1))) \end{aligned} \tag{2}$$

$I(x)$ is the indicator function. It equals to 1 if and only if $x \geq 0$. $I(j) = 1$ means that the candidate which is less 'similar' to the reference is ranked higher than the reference. Intuitively, $Loss(\vec{w})$ is the weighted sum of all the wrongly ranked candidates.

It is difficult to optimize $Loss(\vec{w})$ because $Loss(\vec{w})$ is discontinuous. We make a relaxation here[2]:

$$\begin{aligned} L(\vec{w}) = & \sum_{i=1}^{n}\sum_{j=1}^{k}((s(x_i, y_i^1) - s(x_i, y_i^j)) \\ & * \frac{1}{1 + e^{-\vec{w}^T(\phi(x_i, y_i^j) - \phi(x_i, y_i^1))}}) \\ & \leq \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{k}((s(x_i, y_i^1) - s(x_i, y_i^j)) \\ & * I(\vec{w}^T\phi(x_i, y_i^j) \geq \vec{w}^T\phi(x_i, y_i^1))) \\ & = \frac{1}{2}Loss(\vec{w}) \end{aligned} \tag{3}$$

From the equations above we can see that $2L(\vec{w})$ is the upper bound of our loss function $Loss(\vec{w})$. Therefore we can optimize $L(\vec{w})$ to approximate $Loss(\vec{w})$.

We can use optimization methods like gradient descent to get the $\vec{w}$ that minimize the loss function. Because L is not convex, it may go into a local minimum. In our experiment we held out 10% data as the develop set and try random initialization to decide the initial $\vec{w}$.

4.2 Features for Re-ranking

One advantage of the re-ranking phase is that it can now use features related to candidates. Therefore, we can use a variety of features. We list them as follows.

1. **The coarse-grained ranking score from the graph random walk phase**. From the description of the previous section we know that this score is the probability a 'walker' 'walk' from the full form node to the current candidate. This is a coarse-grained score because it can only use the information of words inside the window. However, it is still informative because in the re-ranking phase we cannot collect this information directly.
2. **The character uni-grams and bi-grams in the candidate**. This kind of feature cannot be used in the traditional character tagging methods.
3. **The language model score of the candidate**. In our experiment, we train a bi-gram language model using Laplace smoothing on the Chinese Gigaword Data[3].
4. **The length of the candidate**. Intuitively, abbreviations tend to be short. Therefore length can be an important feature for the re-ranking.
5. **The degree of ambiguity of the candidate**. We first define the degree of ambiguity d_i of a character c_i as the number of identical words that contain the character. We then define the degree of ambiguity of the candidate as the sum of all d_i in the candidates. We need a dictionary to extract this feature. We collect all words in the PKU data of the second International Chinese Word Segmentation Bakeoff[4].
6. **Whether the candidate is in a word dictionary**. We use the PKU dictionary in feature 5.
7. **Whether all bi-grams are in a word dictionary**. We use the PKU dictionary in feature 5.
8. **Adjacent Variety(AV) of the candidate**. We define the left AV of the candidate as the probability that in a corpus the character in front of the candidate is a character in the full form. For example if we consider the full form "北京大学"(Peking University) and the candidate "京大", then the left AV of "京大" is the probability that the character preceding "京大" is '北' or '京' or '大' or '学' in a corpus. We can similarly define the right AV, with respect to characters follow the candidate.

The AV feature is very useful because in some cases a substring of the full form may have a confusingly high frequency. In the example of "北京大学"(Peking University), an article in the corpus may mention "北京大学"(Peking University) and

[2]To prove this we need the following two inequalities: 1) when $x \geq 0$, $I(x) \leq \frac{2}{1+e^{-x}}$ and 2) $s(x_i, y_i^1) - s(x_i, y_i^j) \geq 0$.

[3]http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2003T09

[4]http://www.sighan.org/bakeoff2005/

“东京大学”(Tokyo University) at the same time. Then the substring “京大学” may be included in the candidate generation phase for “北京大学” with a high frequency. Because the left AV of “京大学” is high, the re-ranker can easily detect this false candidate.

In practice, all the features need to be scaled in order to speed up training. There are many ways to scale features. We use the most intuitive scaling method. For a feature value x, we scale it as $(x-mean)/(max-min)$. Note that for language model score and the score of random walk phase, we scale based on their log value.

5 Experiments

5.1 Dataset and Evaluation metrics

For the dataset, we collect 3210 abbreviation pairs from the Chinese Gigaword corpus. The abbreviation pairs include noun phrases, organization names and some other types. The Chinese Gigaword corpus contains news texts from the year 1992 to 2007. We only collect those pairs whose full form and corresponding abbreviation appear in the same article for at least one time. For full forms with more than one reasonable reference, we keep the most frequently used one as its reference. We use 80% abbreviation pairs as the training data and the rest as the testing data.

We use the top-K accuracy as the evaluation metrics. The top-K accuracy is widely used as the measurement in previous work (Tsuruoka et al., 2005; Sun et al., 2008, 2009; Zhang et al., 2012). It measures what percentage of the reference abbreviations are found if we take the top k candidate abbreviations from all the results. In our experiment, we compare the top-5 accuracy with baselines. We choose the top-10 candidates from the graph random walk are considered in re-ranking phase and the measurement used is top-1 accuracy because the final aim of the algorithm is to detect the exact abbreviation, rather than a list of candidates.

5.2 Candidate List

Table 3 shows examples of the candidates. In our algorithm we further reduce the search space to only incorporate 10 candidates from the candidate generation phase.

K	Top-K Accuracy
1	6.84%
2	19.35%
3	49.01%
4	63.70%
5	73.60%

Table 4: Top-5 accuracy of the candidate generation phase

5.3 Comparison with baselines

We first show the top-5 accuracy of the candidate generation phase Table 4. We can see that, just like the case of using other feature alone, using the score of random walk alone is far from enough. However, the first 5 candidates contain most of the correct answers. We use the top-5 candidates plus another 5 candidates in the re-ranking phase.

We choose the character tagging method as the baseline method. The character tagging strategy is widely used in the abbreviation generation task (Tsuruoka et al., 2005; Sun et al., 2008, 2009; Zhang et al., 2012). We choose the ‘SK’ labeling strategy which is used in Sun et al. (2009); Zhang et al. (2012). The ‘SK’ labeling strategy gives each character a label in the character sequence, with ‘S’ represents ’Skip’ and ‘K’ represents ‘Keep’. Same with Zhang et al. (2012), we use the Conditional Random Fields (CRFs) model in the sequence labeling process.

The baseline method mainly uses the character context information to generate the candidate abbreviation. To be fair we use the same feature set in Sun et al. (2009); Zhang et al. (2012). One drawback of the sequence labeling method is that it relies heavily on the character context in the full form. With the number of new abbreviations grows rapidly (especially in social media like Facebook or twitter), it is impossible to let the model ‘remember’ all the character contexts. Our method is different from theirs, we use a more intuitive way which finds the list of candidates directly from a natural corpus.

Table 5 shows the comparison of the top-5 accuracy. We can see that our method outperforms the baseline methods. The baseline model performs well when using character features (Column 3). However, it performs poorly without the character features (Column 2). In contrast, without the character features, our method (Column 4) works much better than the sequence labeling method.

Full form	Reference	Generated Candidates	#Enum	#Now
国际政治系 (Department of International Politics)	国政系	国政系,政治系,国际政治,国政治,国政,政治,国际	30	7
无核武器国家 (Non-nuclear Countries)	无核国	核国,无核,核武,核国家,无核国,武器国,无核国家,核武器国,无核武器,核武器国家,核武器,国家,武器	62	13
贩卖毒品 (Drug trafficking)	贩毒	卖毒品,贩毒品,贩卖,毒品,贩毒	14	5
长江经济联合发展股份有限公司 (Yangtze Joint River Economic Development Inc.)	长发公司	合股,长发,长发公司,长江公司,长江经济,联合发展,经济发展,经济联合,长江联合发展,长江发展公司,长江经济联合,经济联合发展,长江经济联合发展,股份有限公司,有限公司,长江,公司,股份,联合,经济	16382	20

Table 3: Generated Candidates. #Enum is the number of candidates generated by enumerating all possible candidates. #Now is the number of candidates generated by our method.

When we add character features, our method (Column 5) still outperforms the sequence labeling method.

K	CRF-char	Our-char	CRF	Our
1	38.00%	**48.60%**	53.27%	**55.61%**
2	38.16%	70.87%	65.89%	73.10%
3	39.41%	81.78%	72.43%	81.96%
4	55.30%	87.54%	78.97%	87.57%
5	62.31%	89.25%	81.78%	89.27%

Table 5: Comparison of the baseline method and our method. CRF-char ('-' means minus) is the baseline method without character features. CRF is the baseline method. Our-char is our method without character features. We define character features as the features that consider the characters from the original full form as their parts.

5.4 Performance with less training data

One advantage of our method is that it only requires weak supervision. The baseline method needs plenty of manually collected full-abbreviation pairs to learn a good model. In our method, the candidate generation and coarse-grained ranking is totally unsupervised. The re-ranking phase needs training instances to decide the parameters. However we can use a very small amount of training data to get a reasonably good model. Figure 2 shows the result of using different size of training data. We can see that the performance of the baseline methods drops rapidly when there are less training data. In contrast, when using less training data, our method does not suffer that much.

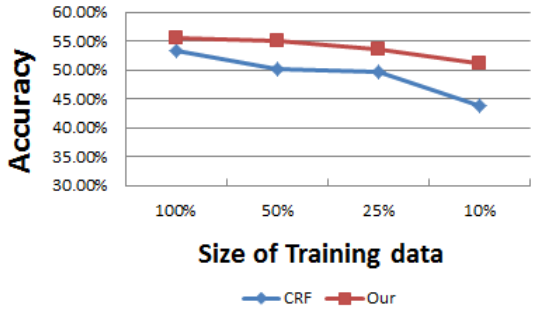

Figure 2: Top-1 accuracy when changing the size of training data. For example, "50%" means "using 50% of all the training data".

5.5 Comparison with previous work

We compare our method with the method in the previous work DPLVM+GI in Sun et al. (2009), which outperforms Tsuruoka et al. (2005); Sun et al. (2008). We also compare our method with the web-based method CRF+WEB in Zhang et al. (2012). Because the comparison is performed on different corpora, we run the two methods on our data. Table 6 shows the top-1 accuracy. We can see that our method outperforms the previous

methods.

System	Top-K Accuracy
DPLVM+GI	53.29%
CRF+WEB	54.02%
Our method	**55.61%**

Table 6: Comparison with previous work. The search results of CRF+WEB is based on March 9, 2014 version of the Baidu search engine.

5.6 Error Analysis

We perform cross-validation to find the errors and list the two major errors below:

1. Some full forms may correspond to more than one acceptable abbreviation. In this case, our method may choose the one that is indeed used as the full form's abbreviation in news texts, but not the same as the standard reference abbreviations. The reason for this phenomenon may lie in the fact that the verification data we use is news text, which tends to be formal. Therefore when a reference is often used colloquially, our method may miss it. We can relieve this by changing the corpus we use.

2. Our method may provide biased information when handling location sensitive phrases. Not only our system, the system of Sun et al. (2009); Zhang et al. (2012) also shows this phenomenon. An example is the case of "香港民主同盟" (Democracy League of Hong Kong). Because most of the news is about news in mainland China, it is hard for the model to tell the difference between the reference "港同盟" and a false candidate "民盟"(Democracy League of China).

   Another ambiguity is "清华大学"(Tsinghua University), which has two abbreviations "清大" and "清华". This happens because the full form itself is ambiguous. Word sense disambiguation can be performed first to handle this kind of problem.

6 Related Work

Abbreviation generation has been studied during recent years. At first, some approaches maintain a database of abbreviations and their corresponding "full form" pairs. The major problem of pure database-building approach is obvious. It is impossible to cover all abbreviations, and the building process is quite laborious. To find these pairs automatically, a powerful approach is to find the reference for a full form given the context, which is referred to as "abbreviation generation".

There is research on using heuristic rules for generating abbreviations Barrett and Grems (1960); Bourne and Ford (1961); Taghva and Gilbreth (1999); Park and Byrd (2001); Wren et al. (2002); Hearst (2003). Most of them achieved high performance. However, hand-crafted rules are time consuming to create, and it is not easy to transfer the knowledge of rules from one language to another.

Recent studies of abbreviation generation have focused on the use of machine learning techniques. Sun et al. (2008) proposed an SVM approach. Tsuruoka et al. (2005); Sun et al. (2009) formalized the process of abbreviation generation as a sequence labeling problem. The drawback of the sequence labeling strategies is that they rely heavily on the character features. This kind of method cannot fit the need for abbreviation generation in social media texts where the amount of abbreviations grows fast.

Besides these pure statistical approaches, there are also many approaches using Web as a corpus in machine learning approaches for generating abbreviations. Adar (2004) proposed methods to detect such pairs from biomedical documents. Jain et al. (2007) used web search results as well as search logs to find and rank abbreviates full pairs, which show good result. The disadvantage is that search log data is only available in a search engine backend. The ordinary approaches do not have access to search engine internals. Zhang et al. (2012) used web search engine information to rerank the candidate abbreviations generated by statistical approaches. Compared to their approaches, our method only uses a fixed corpus, instead of using collective information, which varies from time to time.

Some of the previous work that relate to abbreviations focuses on the task of "abbreviation disambiguation", which aims to find the correct abbreviation-full pairs. In these works, machine learning approaches are commonly used (Park and Byrd, 2001; HaCohen-Kerner et al., 2008; Yu et al., 2006; Ao and Takagi, 2005). We focus on another aspect. We want to find the abbreviation

given the full form. Besides, Sun et al. (2013) also works on abbreviation prediction but focuses on the negative full form problem, which is a little different from our work.

One related research field is text normalization, with many outstanding works (Sproat et al., 2001; Aw et al., 2006; Hassan and Menezes, 2013; Ling et al., 2013; Yang and Eisenstein, 2013). While the two tasks share similarities, abbreviation prediction has its identical characteristics, like the sub-sequence assumption. This results in different methods to tackle the two different problems.

7 Conclusion

In this paper, we propose a unified framework for Chinese abbreviation generation. Our approach contains two stages: candidate generation and re-ranking. Given a long term, we first generate a list of abbreviation candidates using the co-occurrence information. We give a coarse-grained rank using graph random walk to reduce the search space. After we get the candidate lists, we can use the features related to the candidates. We use a similarity sensitive re-rank method to get the final abbreviation. Experiments show that our method outperforms the previous systems.

Acknowledgments

This research was partly supported by National Natural Science Foundation of China (No.61370117,61333018,61300063),Major National Social Science Fund of China(No.12&ZD227), National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), and Doctoral Fund of Ministry of Education of China (No. 20130001120004). The contact author of this paper, according to the meaning given to this role by Key Laboratory of Computational Linguistics, Ministry of Education, School of Electronics Engineering and Computer Science, Peking University, is Houfeng Wang. We thank Ke Wu for part of our work is inspired by his previous work at KLCL.

References

Adar, E. (2004). Sarad: A simple and robust abbreviation dictionary. *Bioinformatics*, 20(4):527–533.

Ao, H. and Takagi, T. (2005). Alice: an algorithm to extract abbreviations from medline. *Journal of the American Medical Informatics Association*, 12(5):576–586.

Aw, A., Zhang, M., Xiao, J., and Su, J. (2006). A phrase-based statistical model for sms text normalization. In *Proceedings of the COLING/ACL on Main conference poster sessions*, pages 33–40. Association for Computational Linguistics.

Barrett, J. and Grems, M. (1960). Abbreviating words systematically. *Communications of the ACM*, 3(5):323–324.

Bourne, C. and Ford, D. (1961). A study of methods for systematically abbreviating english words and names. *Journal of the ACM (JACM)*, 8(4):538–552.

HaCohen-Kerner, Y., Kass, A., and Peretz, A. (2008). Combined one sense disambiguation of abbreviations. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, pages 61–64. Association for Computational Linguistics.

Hassan, H. and Menezes, A. (2013). Social text normalization using contextual graph random walks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1577–1586, Sofia, Bulgaria. Association for Computational Linguistics.

Hearst, M. S. (2003). A simple algorithm for identifying abbreviation definitions in biomedical text.

Jain, A., Cucerzan, S., and Azzam, S. (2007). Acronym-expansion recognition and ranking on the web. In *Information Reuse and Integration, 2007. IRI 2007. IEEE International Conference on*, pages 209–214. IEEE.

Li, J., Ott, M., and Cardie, C. (2013). Identifying manipulated offerings on review portals. In *EMNLP*, pages 1933–1942.

Ling, W., Dyer, C., Black, A. W., and Trancoso, I. (2013). Paraphrasing 4 microblog normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 73–84, Seattle, Washington, USA. Association for Computational Linguistics.

Nenadić, G., Spasić, I., and Ananiadou, S. (2002). Automatic acronym acquisition and term variation management within domain-specific texts.

In *Third International Conference on Language Resources and Evaluation (LREC2002)*, pages 2155–2162.

Norris, J. R. (1998). *Markov chains*. Number 2008. Cambridge university press.

Park, Y. and Byrd, R. (2001). Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 conference on empirical methods in natural language processing*, pages 126–133.

Sproat, R., Black, A. W., Chen, S., Kumar, S., Ostendorf, M., and Richards, C. (2001). Normalization of non-standard words. *Computer Speech & Language*, 15(3):287–333.

Sproat, R., Tao, T., and Zhai, C. (2006). Named entity transliteration with comparable corpora. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 73–80. Association for Computational Linguistics.

Sun, X., Li, W., Meng, F., and Wang, H. (2013). Generalized abbreviation prediction with negative full forms and its application on improving chinese web search. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 641–647, Nagoya, Japan. Asian Federation of Natural Language Processing.

Sun, X., Okazaki, N., and Tsujii, J. (2009). Robust approach to abbreviating terms: A discriminative latent variable model with global information. 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*, pages 905–913. Association for Computational Linguistics.

Sun, X., Wang, H., and Wang, B. (2008). Predicting chinese abbreviations from definitions: An empirical learning approach using support vector regression. *Journal of Computer Science and Technology*, 23(4):602–611.

Taghva, K. and Gilbreth, J. (1999). Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition*, 1(4):191–198.

Tsuruoka, Y., Ananiadou, S., and Tsujii, J. (2005). A machine learning approach to acronym generation. In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 25–31. Association for Computational Linguistics.

Wren, J., Garner, H., et al. (2002). Heuristics for identification of acronym-definition patterns within text: towards an automated construction of comprehensive acronym-definition dictionaries. *Methods of information in medicine*, 41(5):426–434.

Yang, Y. and Eisenstein, J. (2013). A log-linear model for unsupervised text normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 61–72, Seattle, Washington, USA. Association for Computational Linguistics.

Yu, H., Kim, W., Hatzivassiloglou, V., and Wilbur, J. (2006). A large scale, corpus-based approach for automatically disambiguating biomedical abbreviations. *ACM Transactions on Information Systems (TOIS)*, 24(3):380–404.

Zhang, L., Li, S., Wang, H., Sun, N., and Meng, X. (2012). Constructing Chinese abbreviation dictionary: A stacked approach. In *Proceedings of COLING 2012*, pages 3055–3070, Mumbai, India. The COLING 2012 Organizing Committee.

Zhu, X., Ghahramani, Z., Lafferty, J., et al. (2003). Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, volume 3, pages 912–919.