Sun X, Wang HF, Wang B. Predicting Chinese abbreviations from definitions: An empirical learning approach using support vector regression. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 23(4): 602–611 July 2008

Predicting Chinese Abbreviations from Definitions: An Empirical Learning Approach Using Support Vector Regression

Xu Sun[1,2] (孙 栩), Hou-Feng Wang[1] (王厚峰), and Bo Wang[1] (王 波)

[1] *Institute of Computational Linguistics, School of Electronics Engineering and Computer Science, Peking University Beijing 100871, China*

[2] *Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo Tokyo 113-0033, Japan*

E-mail: sunxu@is.s.u-tokyo.ac.jp; wanghf@pku.edu.cn; bowang@pku.edu.cn

Received May 8, 2007; revised April 2, 2008.

Abstract In Chinese, phrases and named entities play a central role in information retrieval. Abbreviations, however, make keyword-based approaches less effective. This paper presents an empirical learning approach to Chinese abbreviation prediction. In this study, each abbreviation is taken as a reduced form of the corresponding definition (expanded form), and the abbreviation prediction is formalized as a scoring and ranking problem among abbreviation candidates, which are automatically generated from the corresponding definition. By employing Support Vector Regression (SVR) for scoring, we can obtain multiple abbreviation candidates together with their SVR values, which are used for candidate ranking. Experimental results show that the SVR method performs better than the popular heuristic rule of abbreviation prediction. In addition, in abbreviation prediction, the SVR method outperforms the hidden Markov model (HMM).

Keywords statistical natural language processing, abbreviation prediction, support vector regression, word clustering

1 Introduction

In Chinese, a specific lexical item that is derived from longer corresponding forms is called "abbreviation". Predicting the abbreviated-forms of the original words or phrases shows that there is potential for improving the performance of information systems. One of its applications for Information Retrieval (IR) is to expand the keywords of a query. When we retrieve documents in an IR system, it is necessary to anticipate the possible abbreviated forms of a phrase or a named-entity when a query is formulated: the retrieval may fail if the query does not include the correct abbreviated forms[1]. For the data of one month's *People's Daily*, only around 20% of the documents containing the keyword "欧洲经济与货币联盟" (European Economic and Monetary Union) can be retrieved using the unabbreviated form, while a significant number of related articles cannot be retrieved simply because the abbreviated form "欧盟" is used in the place of "欧洲经济与货币联盟".

Therefore, we must consider efficient acronym management for generating acronyms from the given definition. As for the methods of recognizing acronym-definition pairs from running text, there are many studies for western languages reporting high performance (e.g., over 96% accuracy and 82% recall)[2−6]. However, the studies on predicting acronyms from definitions are relatively few. To address similar problems in western languages like English, research efforts have been made to construct a database of definition-abbreviation pairs based on running texts of natural language documents[6]. However, the major problem with this database approach is the impossibility of offering a complete coverage, due to the fact that not all natural language documents are publicly available[7].

An alternative solution to this problem is to investigate effective methods of automatic abbreviation prediction. A preliminary investigation of English abbreviation prediction was proposed in [7], which adopts a machine learning approach on the basis of letter information and orthographic features, and it reported a top-1 coverage① of 55.1%.

In comparison with western languages, it is funda-

Regular Paper

Supported by the National Natural Science Foundation of China (Grant Nos. 60473138 and 60675035) and the Beijing Natural Science Foundation (Grant No. 4072012).

① Top-1 coverage shows the percentage of the correct abbreviations being covered if we take the best candidate from the outputs.

mentally different for a computer system to deal with abbreviations in Chinese[8]. Chinese language lacks exterior morphological hints like orthographic features (e.g., capitalization), and Chinese abbreviations have a unique formation style.

Chinese abbreviations are derived through a productive lexical process. Although native speakers may possess a tacit knowledge of the process, it cannot be adequately explained by any linguistic theory[9,10]. Chinese abbreviations are formed in a similar way to how acronyms are derived in English, but not necessarily by combining the initials of a name or a compound word. Normally, a Chinese abbreviation can be obtained by taking any one character from each of the words in a compound/phrase[9]. Take for example 历史/语言/研究所 "Institute of History and Philology at Academia Sinica". The corresponding abbreviation, 史语所, is formed by taking the second, first, and third characters respectively from the three constituents.

The study of Chinese abbreviation prediction is in an early stage. An important preliminary investigation has been made by Chang and Lai's[11], which proposed an HMM model reporting the top-1 coverage of 72% on a dataset. In Chang's model, each word within a definition is assumed to generate at least one character in the derivation of abbreviation, namely, no "word-to-null" mapping is allowed. Since our random sampling shows that more than 30% of factual abbreviations contain "word-to-null" mapping, processing such kinds of cases still remains a study task in Chang's model. For example, 欧盟 is a valid abbreviation from 欧洲/经济/与/货币/联盟 "European Economic and Monetary Union". But currently it is not considered in Chang's model because it contains "word-to-null" mapping (e.g., 经济-to-null).

To find a more general solution to automatic abbreviation prediction, we propose in this paper an empirical learning approach using SVR. In practice, the process of abbreviation prediction is formalized as a scoring and ranking problem among abbreviation candidates. By employing support vector regression for function estimation, we can obtain multiple abbreviation candidates, together with their SVR values.

This paper is structured as follows. In Section 2, the system architecture is described. In Section 3, the features employed in our SVR function estimation are presented. Experimental results are shown and discussed in Section 4, and the paper is concluded in Section 5.

2 Abbreviation Prediction Using SVR

2.1 Problem and Definition

Let X denote a set of Chinese characters. A string $S = c_1c_2\cdots c_n$ over X is a finite sequence of characters, where $c_i \in X$ and $1 \leqslant i \leqslant n$. The length of string S, i.e., the number of characters in the string, is denoted as $|S|$.

Let WL be the Chinese *word_list*. If a string w is contained in WL, this string is called a word. A character c occurring in w is denoted as *c-in-w*, and a word w occurring in a word sequence E is denoted as *w-in-E*.

Definition 1 (Abbreviation Prediction). *Let $E = w_1w_2\cdots w_p$ $(p \geqslant 1)$ be a sequence of Chinese words, and $A = c_1c_2\cdots c_q$ $(q \geqslant 1)$ a character string. If for each c_i-in-A, there exists a word w_j-in-E satisfying c_i-in-w_j and $|A| < |w_1| + |w_2| + \cdots + |w_p|$, A is called an abbreviation candidate of E. For a definition of length L, the upper bound of the number of its abbreviation candidates is $2^L - 2$, and the number of true abbreviations, m, satisfies $0 \leqslant m \leqslant 2^L - 2$. For each abbreviation candidate A given definition E, the probability that A is true abbreviation of E is denoted by the conditional probability $P(A|E)$, and abbreviation prediction referred to in this paper is to estimate $P(A|E)$ by utilizing practical approaches.*

To simplify the problem, we propose in this paper the SVR score $f_{\mathrm{SVR}}(A|E)$ for modeling $P(A|E)$. The major difference between $f_{\mathrm{SVR}}(A|E)$ and $P(A|E)$ is that, $P(A|E)$ needs to be normalized while $f_{\mathrm{SVR}}(A|E)$ does not need to. Also, the problem discussed in this paper is actually a sub-problem isolated from the abbreviation prediction problem in real-world cases. For example, in real-world cases, people may also be interested in extracting definitions from sentences by modeling $P(E)$. In this paper, we suppose that definitions are already retrieved from sentences, therefore $P(E)$ is not the focus. Also, we assume that the number of true abbreviations of a definition satisfies $1 \leqslant m \leqslant 2^L - 2$, say, we do not deal with those definitions which have no abbreviations.

Nevertheless, it should be emphasized that the isolated sub-problem of abbreviation prediction discussed in this paper is definitely non-trivial. As a matter of fact, modeling $P(A|E)$ is a core component for a real abbreviation prediction task on the document level. For example, when apply the model to real problems where the definitions might have no abbreviations, it is possible to fulfill the task via a two-tier process: first, as discussed in this paper, to estimate the conditional probabilities $P(A|E)$ and rank the abbreviation candidates A_i of this E according to the learned probabilities; Second, to determine whether the top-1 ranked candidate A_0 is an acceptable abbreviation by choosing a real-value based threshold (if A_0 is not acceptable, then claim that this definition has no abbreviations). We use

this example to illustrate that, with the core component on abbreviation prediction discussed in this paper, abbreviation prediction problems in real applications can be dealt with sufficient flexibility.

2.2 SVR for Abbreviation Scoring

In this paper, we treat the non-normalized estimation $f_{\mathrm{SVR}}(A|E)$ on $P(A|E)$ as a machine learning problem, and we choose SVR approach to estimate the conditional probabilities. In our approach, each abbreviation candidate A of a definition E is presented by a feature vector $\boldsymbol{F}(\boldsymbol{A}, \boldsymbol{E})$, and an SVR approach is adopted to learn the weights $\boldsymbol{W}$ of features and therefore to estimate the conditional probabilities $P(A|E)$.

SVR estimates a continuous-valued function that encodes the fundamental interrelation between a given input and its corresponding output in the training data. For an abbreviation prediction task, the inputs are abbreviation candidates, and the outputs are the estimated scores.

Given the training dataset, i.e., the set of the eigenvalue distributions $\{\lambda_i\}$ and the corresponding relative powers $\{d_i\}$, the regression function of SVR is given as follows:

$$f(x) = \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) K(x_i, x) + b, \tag{1}$$

where $K(\bullet, \bullet)$ is a Kernel Function. The symbols, α_i^* and α_i, are the optimal solutions to the following optimization problem[13] ($C > 0$ and $\varepsilon \geqslant 0$ are constants), and maximize the following function:

$$\begin{aligned} &-\varepsilon \sum_{i=1}^{l} (\alpha_i^* + \alpha_i) + \sum_{i=1}^{l} (\alpha_i^* - \alpha_i) d_i \\ &-\frac{1}{2} \sum_{i,j=1}^{i} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) K(\lambda_i, \lambda_j) \end{aligned} \tag{2}$$

subject to

$$\begin{aligned} &\sum_{i=1}^{l} (\alpha_i^* - \alpha_i) = 0, \\ &0 \leqslant \alpha_i^*, \quad \alpha_i \leqslant C \quad (i = 1, \ldots, l) \end{aligned} \tag{3}$$

the symbol b is calculated by using the equation

$$b = d_i - \sum_{j=1}^{l} (\alpha_j^* - \alpha_j) K(x_j, x_i) \tag{4}$$

in which i satisfies $0 < \alpha_i^*, \alpha_i < C$.

The trained SVR function can then be used to predict outputs for given inputs that were not included in the training set. This is similar to a neural network, which however is based on the empirical risk minimization. In contrast, SVR introduces structural risk minimization into the regression, thereby achieving global optimization, while a neural network achieves a local minimum[14]. Smola and Schölkopf[15] provide a more detailed introduction to SVR.

2.3 System Design

The main task of abbreviation prediction is to estimate the probable abbreviated forms for given definitions. Here, we discuss how the SVR technique can be applied to the abbreviation prediction task. In our framework, abbreviation prediction contains three steps. First, abbreviation candidates are generated with pruning. Second, in the training phase, each abbreviation candidate is assigned a training value, and a continuous-valued function is trained using SVR. Finally, the trained function is used for predicting abbreviations from new definitions.

In practice, SVR is extended to the fitting of a nonlinear function by employing the kernel trick[15] which allows the original non-linear problem to be reformulated in terms of a kernel function. The reformulated problem is linear and can be solved using linear SVR. We used Chang and Lin's SVR implementation[16].

In using SVR for function estimation, the best choice of kernel function and its parameters may depend on the training set. Among the algorithms generally used for finding the best parameters for support vector regression, we chose K-fold cross validation. We selected the Radial Basis Function (RBF) kernel, and tuned parameters using 5-fold cross validation and grid search as described in [17].

Candidate Pruning. During candidate generation, the abbreviated forms of a definition can be enumerated by arbitrarily retaining some characters and deleting others. For example, 电力网 "Electric Network" has six possible abbreviated forms: 电, 力, 网, 电力, 电网, and 力网. In general, if we have a definition of length L, there could be $2^L - 2$ possible abbreviation candidates (excluding the definition itself and the null string)[11]. It is inefficient to generate all of those candidates for the following SVR modeling. An acceptable compromise is necessary, that is, the candidate generation effort should be restricted to the prospective ones. We then employ two simple pruning heuristics below.

Heuristic 1. *The length of a prospective abbreviation candidate should be less than four characters.*

This simple heuristic comes from the fact that abbreviations are normally created for the purpose of short denotation. The statistics on our experimental dataset containing 3643 abbreviations suggest that long abbreviations (length $\geqslant 4$) are extremely rare ($< 1\%$) in Chinese. On the basis of the pruning, we can successfully reduce the number of candidates from $2^L - 2$ to $\sum_{i=1}^{3} C_L^i$, where C_L^i stands for a combination number of i out of L (normally $L > 3$).

Heuristic 2. *A candidate should not be identical to an existing lexicon word*②.

Take the place name, 高雄 "Kaohsiung", for example. It is often abbreviated as 高, but when the term 高雄/中学 "Kaohsiung High School" is abbreviated, the new abbreviation is 雄中. This can be explained by the fact that the alternative 高中 is an existing lexical item meaning "high school"[10].

SVR Training. Using an illustration, we first briefly define "positive example" and "negative example" within the training phase. For a given definition (e.g., 安全/保卫/人员 "Security-Guarding-Staff"), its abbreviation candidates are generated for training, including a unique "correct" candidate (安保员 in this case) and other "wrong" candidates (保人, 全卫人, etc.). A "correct" candidate is called a "positive example", and a "wrong" candidate is called a "negative example".

During the training phase, each abbreviation candidate should be assigned a real value so that a corresponding ⟨input, output⟩ pair can be formalized for the SVR function estimation. A natural intuition for positive examples is to assign them with a maximum score (e.g., 1.0 in implementation). But, how to assign the training values of those negative examples remains a problem. We tested two solutions below.

Solution 1. All negative examples are simply assigned the identical minimum score (e.g., 0.0).

Solution 2. Negative examples are assigned different values according to their respective "exactitude" (the percentage of correct characters). For example, suppose two characters within 北工学 is correct. Then in the training phase 北工学 will be assigned a value of 0.67 (2/3).

Our experiments show that, though Solution 1 is quite simple, it surpasses Solution 2 slightly on top-1 coverage. Thus, Solution 1 is chosen.

Abbreviation Prediction. Once the continuous-valued function is successfully trained by the SVR model, it can be used to predict outputs for new inputs that were not included in the training set.

Table 1 presents our system outputs of abbreviation prediction for the definition 安全/监察/室 "Security Supervision Room". According to the SVR values, they are automatically ranked and the top ten abbreviation candidates are listed.

Table 1. Predicted Abbreviations for the Definition 安全/监察/室 "Security Supervision Room" Corresponding SVR Values, and Their Ranks

Ranks	SVR Values	Outputs
1	0.93	安监室
2	0.91	安监
3	0.68	全监室
4	0.65	安室
5	0.65	监室
6	0.61	安察室
7	0.43	安察
8	0.40	全室
9	0.31	全察
10	0.25	全察室
⋮	⋮	⋮
ANSWER		安监室

3 Features

The key to the regression is to select discriminative features that effectively capture the distinction between promising abbreviation candidates and "bad" candidates. On the basis of the investigation, two groups of features are employed: definition-abbreviation mapping features and conceptual sequence formation features.

3.1 Definition-Abbreviation Mapping Features

Support vector regression allows us to incorporate diverse types of features. In our system we consider that abbreviations are generated from corresponding definitions via two steps, that is, "*word selection*" and "*character selection*". Take 东视 from 上海/东方/电视台 "Shanghai Dongfang TV" for example. First, during word selection, 东方 and 电视台 are selected while 上海 is neglected. Then, during character selection the character 东 is selected from 东方, and 视 is selected from 电视台; therefore the abbreviation 东视 is finally derived from the definition. In practice, the following feature templates are employed as definition-abbreviation mapping features.

Mapping Pattern. This feature is represented by using bit patterns. For instance, it is suggested in

②An interesting observation is that this is exactly opposite to the formation of acronyms[10]. Because there is capitalization information available in western languages, acronyms with forms identical to existing lexical entries are favored. Examples are SALT, WASP, and WHO.

[11] that many 4-character compounds will be abbreviated to 2-character abbreviations, and many such 4-character words are abbreviated by reserving the first character of each word, which can be represented by a "1010" bit pattern, wherein "1" signifies the reservation of the respective character, and "0" signifies the deletion of it. As an example of "1010", consider the situation where we are going to produce the abbreviation 台大 from 台湾/大学 (Taiwan University).

Word Selection. On the basis of the study of the training data, we notice that in some cases the first word of the definition will not be selected for abbreviation production, particularly when the first word is a location name, e.g., 东视 from 上海/东方/电视台 "Shanghai Dongfang TV" and 纺大 from 中国/纺织/大学 "China Textile University". This feature is employed to determine which type of words tends to be selected for abbreviation generation, and which type does not tend to.

Character Selection. On the basis of the words selected, this feature is employed to find out what kind of character tends to be further selected. Considering the situation where 台大 is produced from 台湾/大学 (Taiwan University), the selected characters are 台 and 大, and the neglected ones are 湾 and 学. In our system, this feature is defined as a string. For example, in the case where 台 is selected from 台湾, the corresponding feature would be defined as "CharSlt_台湾_台" during implementation. A more detailed discussion of "Character Selection" can be found in [12].

3.2 Conceptual Sequence Formation Features

As proposed in Subsection 3.1, abbreviation generation can be treated as a two-tier selection problem that includes word selection and character selection. From a different perspective, however, we can treat it as a conceptual sequence formation problem in which a Chinese abbreviation is generated by another two-tier process: "*Concept Sequence Formalization*" and "*Character Selection*". First, a person chooses a sequence of concepts to set up the concept structure of the abbreviation; then, he attempts to express each concept by choosing characters. We assume that different abbreviations may share a common concept structure. For instance, 上影厂 "Shanghai-Film-Studio" and 北工大 "Beijing University of Technology" are generated from the same concept structure "location + category/industry + entity-postfix". Therefore, although new abbreviations are constantly being created, their concept structure may remain the same. In our system, those concepts are modeled by automatically grouping words into equivalence classes, which are called "word classes" in this paper. Since a word is most often represented as a character in abbreviations, in this paper we could also call it an "abbreviated-word class" or simply a "character class".

3.2.1 Word Clustering

An efficient method for word clustering has been introduced in [18] for machine translation, and its main method is adopted for determining word classes in our study. To automatically group words into equivalence classes and thus attain concepts to formalize our conceptual sequence formation features, we used a maximum-likelihood criterion:

$$C^* = \operatorname*{argmax}_C P(w_1^N|C), \tag{5}$$

where w_1^N represents a token sequence indexed from 1 to N by their positions. A simple approximation of the probability $P(w_1^N|C)$ is to model it as a product of word-class-based bigram probabilities[18]:

$$P(w_1^N|C) = \prod_{i=1}^{N} P(C(w_i)|C(w_{i-1})) \bullet P(w_i|C(w_i)), \tag{6}$$

where the function C maps words w to their class $C(w)$. In this equation, there are two types of probabilities: the transition probability $P(C|C')$ for class C given its predecessor class C', and the membership probability $P(w|C)$ for word w given class C. The transition probability and membership probability are employed to model "Concept Sequence Formalization" and "Character Selection" respectively.

The transition probability $P(C|C')$ and membership probability $P(w|C)$ can be estimated by relative frequencies: $P(C|C') = n(C', C)/n(C')$ and $P(w|C) = n(w)/n(C)$. The function $n(\bullet)$ provides the frequency of a unigram or bigram in the training corpus. If we insert this into (6), we will arrive at the following probability model:

$$P(w_1^N|C) = \prod_{C,C'} n(C, C')^{n(C,C')} \bullet \prod_{w_i} n(w_i)^{n(w_i)} \bullet \Big(\prod_{C} n(C)^{n(C)}\Big)^{-2}. \tag{7}$$

We insert (7) into (5) and take the logarithm, then obtain:

$$\log P(w_1^N|C) = \sum_{w_i} h(n(w_i)) + \sum_{C,C'} h(n(C, C')) - 2\sum_{C} h(n(C)). \tag{8}$$

The function $h(x)$ is a shortcut for $x \bullet \log(x)$. The function $n(\bullet)$ provides the corpus-level frequency in training data.

Since the summation of $h(n(w_i))$ is constant for a fixed training set, it can be dropped to obtain the following decision rule:

$$D(C,n) = \sum_{C,C'} h(n(C,C')) - 2\sum_{C} h(n(C)), \quad (9)$$

$$C^* = \operatorname*{argmax}_{C} D(C,n). \quad (10)$$

We implemented this algorithm on the platform of Visual Studio 2005. While implementing the "argmax" operation in (10), we use the efficient "exchange algorithm"[19]. It is necessary to fix the number of classes in advance, and the detailed experiment upon deciding the number of word classes is presented in Section 4. When 2914 abbreviations were used and the number of word classes was defined as 50, 14 iterations were observed before our word clustering tool achieved the optimal point, with the average time of 25.0 seconds per iteration.

Here two resulting word classes from our clustering algorithm are selected for illustration. One is "班, 办, 部, 场, 厂, 处, 局, 圈, 室, 署, 厅, 系, 校, 院, 站, ···". As we can see, most of the entries are "organization-name-postfixes", which are frequently used as the last character in the formation of abbreviations. Another word class is "埃, 北, 东, 低, 南, 西, 朝, 成, 川, 滇, 韩, 京, 兰, 黎, 柳, 闽, 欧, ···", As illustrated, most of these entries are abbreviated "location names". Those reasonable results show that our word clustering algorithm works fine in a large extent, although we agree that some noise might exist in the results more or less.

3.2.2 Conceptual Sequence Formation Features

Two features are used for conceptual sequence formation analysis as follows.

Conceptual Sequence Formation Probability. For an abbreviation candidate $w_1^N = w_1 \cdots w_N$, its conceptual sequence formation probability can be estimated by using

$$P(w_1^N|C^*) = \prod_{i=1}^{N} P(C^*(w_i)|C^*(w_{i-1})) \bullet P(w_i|C^*(w_i)) \quad (11)$$

where C^* is the optimized distribution of word classes (representing the distribution of concepts) derived from the word clustering algorithm. Since the conceptual sequence formation probability is represented by a real-value, this feature is defined as a real-value-based feature (real-value-based features can be easily integrated into the SVR model during implementation). To deal with data sparseness problem, we adopt Katz back-off strategy. Refer to [20] for detailed presentation for Katz back-off strategy.

Character Penalty. This feature counts the length in characters of the target abbreviation candidate to balance the conceptual sequence formation probability. Without this feature, the final abbreviation produced may be short, because shorter items tend to get higher conceptual sequence formation probability. In our system, this feature is simply defined as an integer representing the length of corresponding abbreviation candidates; for example, for the candidate 北大, its corresponding character penalty value is 2.

4 Evaluation

4.1 Experimental Data

We collected the experimental data based on the book "Modern Chinese-Abbreviation Dictionary"[21] containing 3643 definition-abbreviation pairs. Definitions in the corpus are automatically segmented by employing a word lexicon containing around 100K words. This lexicon contains raw words with no additional information being marked.

The dataset is made up of phrases and named entities, including Noun Phrases (NP), Verb Phrases (VP), Adjective Phrases (AP), Organization Names (ON), Location Names (LN), and Human Names (HN), etc. To study the statistical distribution of each category, we randomly selected 400 pairs from our dataset, and the distribution is shown in Fig.1. This survey may mirror the overall distribution of Chinese abbreviations in a natural way.

The collected dataset is divided into two sets: one with 2914 (80%) pairs, for training (5-fold cross-validation SVR training using RBF kernel function, as presented in Subsection 2.3), and the other containing 729 (20%) pairs, for testing. Word statistics and character statistics for the experimental data are shown in Table 2. Although the total numbers of word tokens and character tokens are kind of huge, we can notice from the statistics that the word vocabulary and the character list have a much smaller size. Since the Word Selection and Character Selection features are built based on the word vocabulary and character list, it is possible to train out statistically reliable weights for those Word Selection and Character Selection features even based on a relatively small size of training data. The word classes for conceptual sequence formation features are learned from the training data.

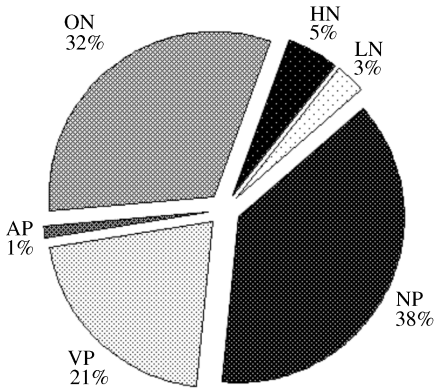


Fig.1. Distribution of Chinese abbreviations: a random sampling of 400 abbreviations in our collection, including Noun Phrases (NP), Verb Phrases (VP), Adjective Phrases (AP), Organization Names (ON), Location Names (LN), and Human Names (HN), etc.

Table 2. Characteristics of Words and Characters of the Experimental Data Set

Definitions	Words	7 663
	Vocabulary	2 664
	Characters	14 587
	Character List	1 590
Abbreviations	Characters	6 326
	Character List	1 206

4.2 Determining the Number of Word Classes

To a large extent, the performance of conceptual sequence formation features depends on the quality of word classes, and the quality of word classes depends on our word clustering algorithm. When we implement the word clustering algorithm, it is necessary to fix the number of classes in advance; otherwise, the optimum will be reached if every word is a class of its own.

Consequently, we performed an experiment to determine the optimal number of word classes, and the experimental results are shown in Fig.2. As can be noticed, the curves show that choosing 50 classes is optimal for top-1 and top-2 coverage, with coverage rates of 63% and 80% respectively. In view of these experimental results, we chose 50 as the default number of word classes during word clustering.

4.3 Experimental Evaluation

In the training phase we ran a 5-fold cross-validation SVR training using an RBF kernel function. The dataset for training consisted of 2914 definition-abbreviation pairs.

As has been suggested in [7], the performance indicators used in this paper are top-1, top-2 and top-3 coverage. Top-N coverage shows the percentage of the correct abbreviations covered if we take top N candidates from the outputs of the abbreviation generator.

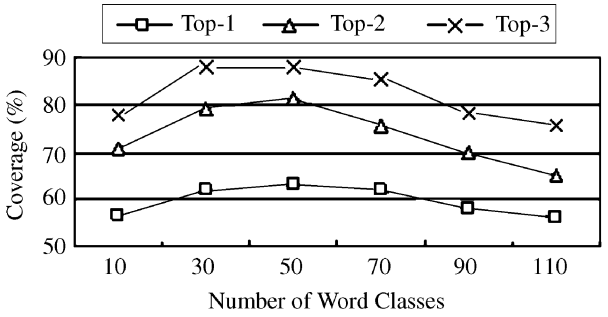


Fig.2. Experimental results on various numbers of word classes (curves of top-1, top-2 and top-3 coverage).

We conducted incrementally the following five experiments:

1) SVR approach using Mapping Pattern (MP) features;
2) integrating Word Selection (WS) features into 1);
3) integrating Character Selection (CS) features into 2);
4) integrating Conceptual Sequence Formation (CF) features with 3);
5) integrating Length Penalty (LP) features with 4).

Table 3. Incremental Evaluation of the Abbreviation Prediction System

Feature Templates	No. of Def.	No.-of-Outputs/Def.	Top-1 Coverage (%)	Top-2 Coverage (%)	Top-3 Coverage (%)
MP			42.5	60.8	65.0
MP, WS			51.0	69.0	81.3
MP, WS, CS	729	17	60.9	79.1	86.1
MP, WS, CS, CF			62.7	80.3	87.6
MP, WS, CS, CF, LP			62.7	80.4	87.7

The details of incremental evaluation are shown in Table 3. Notice that, on average, the number of abbreviation candidates from a definition is 76 candidates, and after the pruning from the two heuristics proposed in Subsection 2.3, the average number of system outputs (abbreviation candidates) per definition was reduced to 17.

On the one hand, we consider that abbreviations are generated from corresponding definitions via Word Selection and Character Selection. As we have shown, WS and CS features prove to be the critical features which cause substantial increases of top-N coverage. Such considerable improvements show that, to some extent, the process of word selection and character selection within Chinese abbreviation generation is well-regulated and can be statistically learned. Take character selection for instance, for the word 突袭 "assault" in a given definition, normally 袭 would be more preferred than 突. Experimental results indicate that well-regulated selections can be successfully learned by the SVR model.

On the other hand, in our system we consider abbreviation generation as a conceptual sequence formation problem, and proposed Conceptual Sequence Formation features for better ranking abbreviation candidates.

The integration of Conceptual Sequence Formation features has led to better results, especially on top-1 coverage. This improvement suggests that the conceptual sequence formation scoring is workable, and the algorithm for word clustering is capable of building qualified word classes for conceptual modeling. However, we also realize that, in comparison with WS and CS features, the improvement from CF features is not that significant. This is due to the fact that, to successfully train such CF features, a large training dataset is required. However, our current collection is probably not adequate (merely 2914 pairs). We believe that it can improve if we collect a larger training set in the future.

The integration of Length Penalty features yields a slightly better top-2 and top-3 coverage. Yet the improvement is quite marginal, and the primary reason may be that the length information of an abbreviation candidate has already been contained implicitly in its Mapping Pattern features. In this case the additional information carried by the LP features is quite limited.

4.4 Related Work and Comparison

One of the simplest ways to generate abbreviations from definitions is to choose the characters at the beginning of each word. For comparison, we tested the performance of this popular heuristic function, as is shown on the top row in Table 4.

With the top-1 coverage of 62.7%, our system outperformed the standard heuristic rule reaching top-1 coverage of 41.6%. Note that, since the heuristic function predicts only one abbreviation candidate for each definition, only the top-1 coverage is available.

Looking back on Table 3, we notice that the SVR approach using solely MP features has already outperformed the popular heuristic. As a matter of fact, from a different viewpoint, the popular heuristic can be deemed as a special Mapping Pattern, and MP features as a whole may stand for a generalized heuristic. For example, consider 台大 from 台湾/大学 (Taiwan University), the popular heuristic choosing characters at the beginning of each word can be represented by using the corresponding Mapping Pattern feature "1010".

In the field of empirical learning, an important system on Chinese abbreviation prediction is Chang's HMM model proposed in [11]. Here we briefly introduce Chang's model: the best possible abbreviation form c^* for an input word compound w can be determined as the one with the highest generation probability $P(c|w)$, i.e., $c^* = \text{argmax}_c P(c|w)$. The generation probability for a candidate c, in turn, can be estimated by summing up all probabilities of alignments between each character c_j in c and the suspect constituent words w_j in w, which goes to

$$P(c|w) = \sum_{A \in \text{all-alignments}} P(c, A|w) \equiv \sum_{A} P_A(c|w) \tag{12}$$

where $P_A(c|w)$ is the generation probability for a known alignment A. This can be estimated by:

$$P(c|w) = P(c_i^j|w_m^n) = \prod_{k=1, j-i+1} P(c_{i,k}|w_{m,k}) \bullet P(w_{m,k}|w_{m,k-1}). \tag{13}$$

Table 4. Comparison Among the Popular Heuristic, HMM Model and the SVR Prediction Model

Models	No. of Def.	No.-of-Outputs/Def.	Top-1 Coverage (%)	Top-2 Coverage (%)	Top-3 Coverage (%)
Simple Heuristic			41.6	–	–
HMM	729	17	46.1	–	–
SVR			62.7	80.4	87.7

As presented in Section 1, a limit to Chang's model is that complicated prediction containing word-to-null mapping cannot be handled properly in current phase. For this reason, in [11] experiments were performed on definition-abbreviation pairs that contain no word-to-null mapping, and a top-1 coverage of 72% was achieved.

Since Chang's system is not openly available, we implemented an HMM system based on [11], including the features: membership probability $P(c_i|w_m)$, word transition probability $P(w_m|w_{m-1})$, abbreviation pattern probability $P(bit|n)$, and length feature $P(m|n)$.

The performance of the HMM model (our implementation) is presented in Table 4. As suggested by the experimental results, the SVR system outperforms the HMM system, when trained and tested on the same dataset. This dataset contains word-to-null mapping, which makes the original HMM system hard to process it (we agree that it may not be difficult to adopt some strategy to free the strict model assumption in HMM-based abbreviation predictor. However, the discussion in this direction is not the focus of this paper). Our SVR model can robustly deal with word-to-null mapping in abbreviation prediction, while the HMM model may need improvement. In fact, among the 3643 pairs of our experimental dataset, 1362 pairs (37%) contain word-to-null mapping, and this made the experimental difference.

5 Conclusion and Future Work

An empirical learning approach to Chinese abbreviation prediction was presented. This proposal may be helpful for applications regarding information retrieval, such as query expansion. It is also closely related to abbreviation identification[22] and abbreviation resolution study[23].

In this approach, we take the prediction process as a scoring and ranking problem based on SVR modeling, so that abbreviation candidates are predicted together with their function values representing their likelihood. Experiments were performed by using 729 definition-abbreviation pairs, and encouraging results were achieved; the results are relatively better than those obtained by the popular heuristics and by the HMM-based prediction model.

While other features are not proven to be predictive in the current environment, some of them may well be predictive in the future with better parameter tuning and a larger training data. Huang[9,10] suggested that collocation-based mutual information may be helpful for abbreviation prediction.

Moreover, in this paper we did not consider the generation mechanism where the characters in the abbreviation appear in a different order in the definition (a small part of abbreviations involve this generation mechanism). We may further improve the performance by taking such permutation of characters into consideration.

Acknowledgement Xiang Ma helped with software implementation on Linux. The authors also wish to thank the anonymous referees for their suggestions for improving this paper.

References

[1] Wren J D, Chang J T, Pustejovsky J, Adar E, Garner H R, Altman R B. Biomedical term mapping databases. *Nucleic Acid Research,* 2005, 33: 289–293.

[2] Yoshida M, Fukuda K, Takagi T. Pnad-css: A workbench for constructing a protein name abbreviation dictionary. *Bioinformatics,* 2000, 16(2): 169–175.

[3] Nenadic G, Spasic I, Ananiadou S. Automatic acronym acquisition and term variation management within domain-specific texts. In *Proc. the LREC-3,* Las Palmas, Spain, 2002, pp.2155–2162.

[4] Schwartz A, Hearst M. A simple algorithm for identifying abbreviation definitions in biomedical texts. In *Proc. the Pacific Symposium on Biocomputing* (*PSB 2003*), pp.451–462.

[5] Manuel Zahariev. An efficient methodology for acronym-expansion matching. In *Proc. the International Conference on Information and Knowledge Engineering* (*IKE*), Las Vegas, USA, 2003, pp.32–37.

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

[7] Tsuruoka Y, Ananiadou S, Tsujii J. A machine learning approach to abbreviation generation. In *Proc. the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics,* Michigan, USA, 2005, pp.25–31.

[8] Fu G, Luke K, Zhang M, Zhou G. A hybrid approach to Chinese abbreviation expansion. In *Proc ICCPOL'06: 21st International Conference on Computer Processing of Oriental Languages,* Singapore, 2006, pp.277–287.

[9] Huang C R, Ahrens K, Chen K J. A data-driven approach to psychological reality of the mental lexicon: Two studies on Chinese corpus linguistics. In *Proc. Language and Its Psychobiological Bases,* Taipei, 1994a.

[10] Huang C R, Hong W M, Chen K J. Suoxie: An information based lexical rule of abbreviation. In *Proc. the Second Pacific Asia Conference on Formal and Computational Linguistics II,* Japan, 1994b, pp.49–52.

[11] Chang J, Lai L. A preliminary study on probabilistic models for Chinese abbreviations. In *Proc. the Third SIGHAN Workshop on Chinese Language Learning,* ACL, Barcelona, Spain, 2004, pp.9–16.

[12] Chang J, Teng T. Mining atomic Chinese abbreviation pairs: A probabilistic model for single character word recovery. *Language Resources and Evaluation,* 2007, 40(3/4): 367–374.

[13] Christianini N, Shawe-Taylor J. An Introduction to Support Vector Machines and Other Kernel-Based Methods. Cambridge University Press, 2000.

[14] Eubank R L. Spline Smoothing and Nonparametric Regression. New York: Marcel Dekker, 1988.

[15] Smola A, Schölkopf B. A tutorial on support vector regression. *Statistics and Computing,* 2003, 14(3): 199–222.

[16] Chang C C, Lin C J. LIBSVM: A library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjin/libsvm.

[17] Hsu C W, Chang C C, Lin C J. A Practical Guide to Support Vector Classification, 2003, Working Paper, http://www.csie.ntu.edu.tw/~cjlin/talks/freiburg.pdf.

[18] Och F J. An efficient method for determining bilingual word classes. In *Proc. Ninth Conference of the European Chapter of the Association for Computational Linguistics,* EACL'99, 1999, pp.71–76.

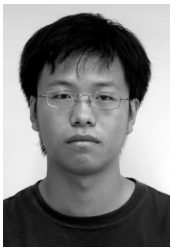
[19] Martin S, Liermann J, Ney H. Algorithms for bigram and trigram word clustering. *Speech Communication,* 1998, 24(1): 19–37.

[20] Katz S M. Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Trans. Acoustics, Speech, and Signal Processing,* 1987, 35(3): 400–401.

[21] Yan H, Wan X. Modern Chinese Abbreviation Dictionary. China: Yuwen Publisher, 2002. (In Chinese)

[22] Sun X, Wang H F. Chinese abbreviation identification using abbreviation-template features and context information. In *Proc. 21st International Conference on Computer Processing of Oriental Languages* (*ICCPOL-06*), Singapore, 2006, pp.245–255.

[23] Sun X, Wang H F, Zhang Y. Chinese abbreviation-definition identification: A SVM approach using context information. In *Proc. PRICAI-06: the 9th Pacific Rim International Conference on Artificial Intelligence,* 2006, pp.495–504.



Xu Sun is currently a Ph.D. candidate in computer science in The University of Tokyo. He received the M.S. degree from Peking University in 2007 and the B.E. degree from Huazhong University of Science & Technology in 2004, both in computer science. He was also a visiting student in MSR Asia in 2005 fall. His research interests focus on statistical natural language processing, machine learning, and numerical optimization for large-scale computing.

Hou-Feng Wang received his B.S. and Ph.D. degrees from Wuhan University, in 1986 and 1998 respectively, and his M.S. degree from Institute of Computing Technology, Chinese Academy Sciences in 1989, all in computer science. His research interests include natural language processing and machine learning. He is a senior member of China Computer Federation.

Bo Wang is now an M.S. candidate in computer science in Peking University. He received a B.S. degree in computation science from Northern Jiaotong University. He will join Microsoft Research Asia after graduation.