Learning Abbreviations from Chinese and English Terms by Modeling Non-Local Information

XU SUN, Peking University
NAOAKI OKAZAKI, Tohoku University
JUN'ICHI TSUJII, Microsoft Research Asia
HOUFENG WANG, Peking University

The present article describes a robust approach for abbreviating terms. First, in order to incorporate non-local information into abbreviation generation tasks, we present both implicit and explicit solutions: the latent variable model and the label encoding with global information. Although the two approaches compete with one another, we find they are also highly complementary. We propose a combination of the two approaches, and we will show the proposed method outperforms all of the existing methods on abbreviation generation datasets. In order to reduce computational complexity of learning non-local information, we further present an online training method, which can arrive the objective optimum with accelerated training speed. We used a Chinese newswire dataset and a English biomedical dataset for experiments. Experiments revealed that the proposed abbreviation generator with non-local information achieved the best results for both the Chinese and English languages.

Categories and Subject Descriptors: C.2.2 [**Natural Language Processing**]: Text Analysis

General Terms: Experimentation, Algorithms, Languages

Additional Key Words and Phrases: Abbreviation processing, non-local information, machine learning, stochastic learning

ACM Reference Format:
Sun, X., Okazaki, N., Tsujii, J., and Wang, H. 2013. Learning abbreviations from Chinese and English terms by modeling non-local information. *ACM Trans. Asian Lang. Inform. Process.* 12, 2, Article 5 (June 2013), 17 pages.
DOI:http://dx.doi.org/10.1145/2461316.2461317

1. INTRODUCTION

Abbreviations represent fully expanded forms (e.g., Hidden Markov Model) through the use of shortened forms (e.g., HMM). Abbreviations result from a highly productive type of term variation, defining alternative expressions to fully expanded forms. At the same time, abbreviations increase the ambiguity in a text. For example, in computational linguistics, the acronym HMM stands for Hidden Markov Model, whereas,

This work is an extended version of the conference article Sun et al. [2009b].

This work was partially supported by JST, PRESTO. This work was also partially supported by the National High Technology Research and Development Program of China (863 Program) (No. 2012AA011101), the National Natural Science Foundation of China (No. 91024009, No. 60973053, No. 90920011), and the Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 20090001110047).

Authors' addresses: X. Sun, Key Laboratory of Computational Linguistics (Peking University), Ministry of Education, China; email: xusun@pku.edu.cn; N. Okazaki, Graduate School of Information Sciences, Tohoku University, Japan; J. Tsujii, Microsoft Research Asia, Beijing, China; H. Wang, Key Laboratory of Computational Linguistics (Peking University), Ministry of Education, China.

DOI:http://dx.doi.org/10.1145/2461316.2461317

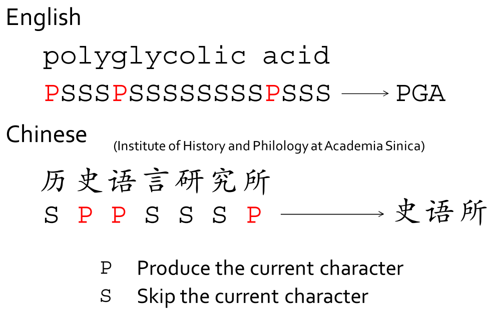

English
polyglycolic acid
PSSSPSSSSSSSSSPSSS → PGA
Chinese
(Institute of History and Philology at Academia Sinica)
历史语言研究所
S P P S S S P → 史语所
P Produce the current character
S Skip the current character

Fig. 1. English and Chinese abbreviation generation as a sequential labeling problem.

in the field of biochemistry, HMM is generally an abbreviation for heavy meromyosin. Associating abbreviations with their fully expanded forms is of great importance in various natural language processing (NLP) applications [HaCohen-Kerner et al. 2008; Pakhomov 2002; Yu et al. 2006].

The core technology for abbreviation disambiguation is to recognize the abbreviation definitions in the actual text. Chang and Schütze [2006] reported that 64,242 new abbreviations were introduced into the biomedical literatures in 2004. As such, it is important to maintain sense inventories (lists of abbreviation definitions) that are updated with the newly coined word or phrase. In addition, based on the one-sense-per-discourse assumption [Yarowsky 1993], the recognition of abbreviation definitions assumes senses of abbreviations that are locally defined in a document. Therefore, a number of studies have attempted to model the generation processes of abbreviations, for example, inferring the abbreviating mechanism of the Hidden Markov Model into HMM.

An obvious approach is to manually design rules for abbreviating terms. Early studies attempted to determine the generic rules that humans use to intuitively abbreviate given words [Barrett and Grems 1960; Bourne and Ford 1961]. Since the late 1990s, researchers have presented various methods by which abbreviation definitions appearing in actual texts are extracted [Adar 2004; Ao and Takagi 2005; Park and Byrd 2001; Schwartz and Hearst 2003; Taghva and Gilbreth 1999; Wren and Garner 2002]. These studies proposed various heuristics for abbreviation recognition, for example, use of initials, capital letters, syllable boundaries, stop words, length of abbreviation, etc. These studies performed highly, especially for English abbreviations. However, a more extensive investigation of abbreviations is needed in order to further improve definition extraction. In addition, we cannot simply transfer the knowledge of the hand-crafted rules from one language to another. For instance, in English, abbreviation characters are preferably chosen from the initial and/or capital characters in their full forms, whereas some other languages, including Chinese and Japanese, do not have word boundaries or case sensitivity.

A number of recent studies have investigated the use of machine learning techniques. For example, Tsuruoka et al. [2005] formalized the processes of abbreviation generation as a sequence labeling problem. In the present study, each character in the expanded form is tagged with a label, $y \in \{\mathtt{P}, \mathtt{S}\}$[1], where the label P *produces the current character* and the label S *skips the current character*. In Figure 1 (upper), the abbreviation *PGA* is generated from the full form *polyglycolic acid* because the underlined characters are tagged with P labels. In Figure 1 (lower), the abbreviation is generated

[1] Although the original article by Tsuruoka et al. [2005] attached case sensitivity information to the P label, for simplicity, we herein omit this information.

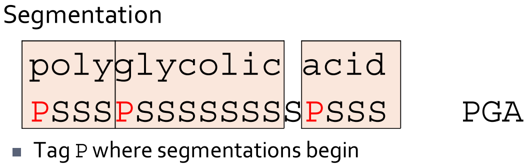


Fig. 2. Abbreviation generation could be more reasonable if it is able to "segment" a word: local information is not enough here.

using the second and third characters, skipping the subsequent three characters, and then using the seventh character.

In order to formalize this task as a sequential labeling problem, we have assumed that the label of a character is determined by the local information of the character and its previous label. However, this assumption has a limitation for modeling abbreviations, and adding the global information addresses this limitation. For example, the model cannot make use of the number of words in a full form to determine and generate a suitable number of letters for the abbreviation. In addition, the model would be able to recognize the abbreviating process in Figure 1 more reasonably if it is able to segment the word *polyglycolic* into smaller regions, for example, *poly-glycolic* (see Figure 2). In this article, non-local information means the long range information that is located outside of the local observations. For example, for the labeling on the segment *poly* in Figure 2, the labeling on the word *acid* is a type of non-local information. Modeling non-local information can enable long-range dependencies in abbreviating words. Even though humans may use global or non-local information to abbreviate words, previous studies have rarely incorporated this information into a sequential labeling model.

In the present article, we propose implicit and explicit solutions for incorporating non-local information. The implicit solution is based on the latent conditional random field model (LCRF) [Sun and Tsujii 2009; Sun et al. 2008b], in which non-local information is modeled by latent variables. We manually encode non-local information into the labels in order to provide an explicit solution. We evaluate the models on the task of abbreviation generation, in which a model produces an abbreviation for a given full form. Experimental results indicate that the proposed models outperform previous abbreviation generation studies.

Not surprisingly, non-local information increases the computational complexity of training and slows down the training speed. In particular, training LCRFs is quite challenging. Traditional batch training methods are very slow on training LCRFs [Petrov and Klein 2008; Sun et al. 2009a]. For example, Petrov and Klein [2008] highlighted both time and memory cost problems on training LCRFs for natural language parsing. To accelerate training speed, we further present an online training method for optimizing LCRFs. The improved online training method can arrive the same objective optimum with significantly accelerated training speed. Online optimization methods for non-convex log-linear models are interesting because few existing studies applied online training for LCRFs.

2. RELATED WORK

Abbreviation generation (i.e., predicting abbreviation of a given full form) is important in Chinese NLP applications. For example, in an information retrieval (IR) system, it will be helpful if we can estimate abbreviations of a query. For the data of People's Daily in Chinese, a large number of the articles contain only the abbreviations. Hence, successful abbreviation generation may improve the recall of IR systems. In addition, [Yang et al. 2009] showed that Chinese abbreviation generation can be used for vocabulary expansion and can improve the performance of Chinese voice search.

Abbreviation generation is closely related to abbreviation recognition [Sun et al. 2009b], and abbreviation recognition aims at extracting abbreviation-definition candidate pairs from texts and then get correct abbreviation-definition pairs. We will introduce related work on both abbreviation generation and recognition. Compared with abbreviation recognition, there was much less previous work on abbreviation generation. Early studies on abbreviation generation/recognition attempted to define the generic rules that humans abbreviate given words [Barrett and Grems 1960; Bourne and Ford 1961]. For example, Barrett and Grems [1960] defines 20 rules for abbreviating terms; for example, “Always save the first letter for each single word and the first letter for each phrase.” However, it was unrealistic to design a set of generic rules that are applicable to any abbreviation definitions on different text corpora.

Since the late 1990s, researchers have presented various methods by which abbreviation definitions appearing in actual texts are extracted [Torii et al. 2007]. Most studies share pattern (1) to locate a textual fragment with an abbreviation and its expanded form [Schwartz and Hearst 2003; Wren and Garner 2002].

expanded form “(‘*abbreviation*’)” (1)

Assuming we take $(l + 4)$ words appearing before the parenthetical expression [Adar 2004], where l is the number of letters in the short form, the sentence, “The exact route was determined by magnetic resonance imaging (MRI)”, could yield the textual fragment marked with the italic letters. The most challenging part of this task is to identify the boundary of an expanded form in the textual fragment if exists.

Existing methods of abbreviation recognition can be categorized into three groups: using hand-crafted heuristics and/or scoring rules [Adar 2004; Ao and Takagi 2005; Pakhomov 2002; Park and Byrd 2001; Schwartz and Hearst 2003; Taghva and Gilbreth 1999; Wren and Garner 2002; Yamamoto et al. 2011; Yu et al. 2002], statistics [Hisamitsu and Niwa 2001; Liu and Friedman 2003; Okazaki and Ananiadou 2006; Zhou et al. 2006], and machine learning [Chang and Schütze 2006; Kuo et al. 2009; Nadeau and Turney 2005; Okazaki et al. 2008; Xu and Huang 2006; Yeganova et al. 2010].

The first category proposed a variety of characteristics to identify abbreviation definitions, for example, use of initials, capital letters, syllable boundaries, stop words, length of abbreviation, etc. Schwartz and Hearst [2003] implemented a simple algorithm that obtains the shortest expression containing all alpha-numerical letters of an abbreviation. Adar [2004] presented scoring rules to choose the most likely expanded form in multiple candidates, for example, “add one to the score for every abbreviation character that is at the start of a definition word.” Ao and Takagi [2005] designed more detailed (two-pages long in their article) conditions for accepting or discarding candidates of abbreviation definitions. However, hand-crafted rules are fragile to incorrect abbreviation definitions, for example, *a patient with human immunodeficiency syndrome (AIDS)*. Hand-crafted rules also encounter difficulties in finding an expanded form whose abbreviation is arranged in a different word order, for example, *beta 2 adrenergic receptor (ADRB2)*.

The second category focuses on the redundancy (repetition) of abbreviation definitions in a corpus. Hisamitsu and Niwa [2001] proposed a method that measures the co-occurrence strength between the inner and outer phrases of a parenthetical expression via mutual information, χ^2 test with Yate’s correction, Dice coefficient, log-likelihood ratio, etc. Liu and Friedman [2003] based their method on collocations occurring before the parenthetical expressions. Enumerating candidates of expanded forms as collocations appearing more than once in a text collection, their method eliminates unlikely candidates with rules such as “remove a set of candidates T_w formed by adding a prefix word to a candidate w if the number of such candidates T_w is greater than 3”. Okazaki and Ananiadou [2006] formalized the task of abbreviation recognition to extract technical terms appearing before parenthetical expressions. They used a variant of C-value method [Frantzi and Ananiadou 1999] to score candidates of expanded forms.

To improve the accuracy of abbreviation recognition, the third category obtains abbreviation rules by using a machine learning algorithm. The most popular approach is to classify or score each candidate of expanded forms by defining features that assess the appropriateness of the abbreviation definition. Chang and Schütze [2006] applied a logistic regression to combine nine features including “the percentage of letters of an expanded form aligned at the beginning of a word,” “the percentage of abbreviation letters aligned to the expanded form,” etc. Nadeau and Turney [2005] employed Support Vector Machine (SVM) to classify candidates of abbreviation definitions into positive (definition) or negative (non-definition). The feature design is similar to that of Chang and Schütze [2006], for example, “the number of participating definition letters that are capitalized,” and “the length (in words) of the definition.” Xu and Huang [2006] also presented a similar method using SVM. Because these studies used a small number of features that roughly correspond to the scoring rules used in the first category, these studies did not yield significantly better results than those with hand-crafted rules.

Recently, Kuo et al. [2009] proposed a set of rich features that represent the literal information, character properties, positional information, existence of stop words, recognition results from hand-crafted rules, etc. They reported that their machine learning method outperformed all baseline rule-based systems. Yeganova et al. [2010] explored to use “naturally labeled data”, in which positive instances are naturally occurring potential abbreviation definition pairs in text, and negative instances are generated by randomly mixing potential abbreviations with unrelated potential definitions.

The previous studies presented so far focused only on global features that characterize an expanded form and abbreviation as a whole, for example, the number of abbreviation letters that appear at the head of a full form. In other words, these studies did not model associations between abbreviation letters and their origins explicitly. In contrast, Okazaki et al. [2008] formalize the task of abbreviation recognition as a sequential alignment problem, which finds the optimal alignment (origins of abbreviation letters) between an abbreviation and a full form. They designed a large number features that directly express the events where letters produce or do not produce abbreviations. This approach also outperformed previous studies, but did not incorporate global information (e.g., number of abbreviation letters) to the feature set because the method assumes Markov assumption on the events producing abbreviation letters.

We close this section by describing the previous work on abbreviation generation, which predicts an abbreviation for a given expanded form. Compared to the well-studied abbreviation recognition task, there was much less prior work on abbreviation generation. Abbreviation generation is more difficult than abbreviation recognition, having broader search space (2^n candidates for a given expanded form of n letters) than that for abbreviation recognition (usually w candidates for a window of w words). Basically, abbreviation generation is formalized as a sequential labeling problem: each character in the expanded form is tagged with a label, $y \in \{\mathtt{P}, \mathtt{S}\}$, where $\mathtt{P}$ *produces the current character* and the label $\mathtt{S}$ *skips the current character*. Tsuruoka et al. [2005] used Maximum Entropy Markov Model (MEMM), and Wakaki et al. [2009] used Conditional Random Fields (CRFs) for modeling the generation processes of abbreviations. Although these efforts have been made to advance the approach for abbreviation generation, abbreviation recognition generally yields better performance than generation.

3. ABBREVIATOR WITH NON-LOCAL INFORMATION

3.1. Review of Conditional Random Fields

Conditional Random Fields are proposed as an alternative solution for structured classification by solving “the label bias problem” [Lafferty et al. 2001]. Assuming a feature function that maps a pair of observation sequence x and label sequence y to a global feature vector f, the probability of a label sequence y conditioned on the observation sequence x is modeled as the following [Lafferty et al. 2001].

$$P(\mathbf{y}|\mathbf{x},\mathbf{w}) = \frac{\exp\left[\mathbf{w}^{\top}\mathbf{f}(\mathbf{y},\mathbf{x})\right]}{\sum_{\forall\mathbf{y}'}\exp\left[\mathbf{w}^{\top}\mathbf{f}(\mathbf{y}',\mathbf{x})\right]}, \tag{2}$$

where w is a parameter vector.

Typically, computing $\sum_{\forall\mathbf{y}'}\exp\left[\mathbf{w}^{\top}\mathbf{f}(\mathbf{y}',\mathbf{x})\right]$ could be computationally intractable. This summation can be computed using dynamic programming techniques [Lafferty et al. 2001]. To make the dynamic programming techniques applicable, the dependencies of labels are chosen to obey the Markov property. This has a computational complexity of $O(NK^M)$. N is the length of the sequence, K is the dimension of the label set, and M is the length of the Markov order used by local features.

Given a training set consisting of n labeled sequences, $(\mathbf{x}_i, \mathbf{y}_i)$, for $i = 1 \ldots n$, parameter estimation is performed by maximizing the objective function

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{n} \log P(\mathbf{y}_i|\mathbf{x}_i,\mathbf{w}) - R(\mathbf{w}). \tag{3}$$

The first term of this equation represents a conditional log-likelihood of a training data. The second term is a regularizer for reducing overfitting. We employed an L_2 prior, $R(\mathbf{w}) = \frac{||\mathbf{w}||^2}{2\sigma^2}$. In what follows, we denote the conditional log-likelihood of each sample, $\log P(\mathbf{y}_i|\mathbf{x}_i,\mathbf{w})$, as $\ell(i,\mathbf{w})$. The final objective function is the following.

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{n} \ell(i,\mathbf{w}) - \frac{||\mathbf{w}||^2}{2\sigma^2}. \tag{4}$$

3.2. A Latent Variable Abbreviator

Real-world problems may contain hidden structures that are difficult to be captured by conventional structured classification models without latent variables. In such cases, models that exploit latent variables are advantageous in learning [Morency et al. 2007; Petrov and Klein 2008; Sun et al. 2009a, 2009b]. Therefore, as a representative structured classification model with latent variables, the latent conditional random fields have become popular for performing a variety of tasks with hidden structures, for example, vision recognition [Morency et al. 2007], syntactic parsing [Petrov and Klein 2008], abbreviation generation/recognition [Sun et al. 2009b], and biomedical named entity recognition [Sun et al. 2009a]. For example, Morency et al. [2007] demonstrated that LCRF models can learn latent structures of vision recognition problems efficiently, and outperform several widely-used conventional models, such as support vector machines, conditional random fields and hidden Markov models (HMMs). Petrov and Klein [2008] reported on a syntactic parsing task that LCRF models can learn more accurate grammars than that of the conventional techniques without latent variables.

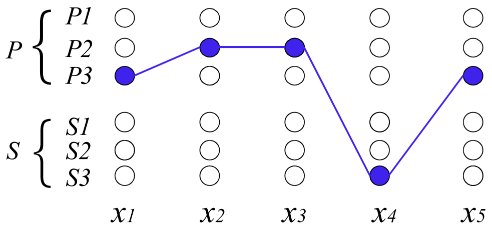


Fig. 3. An example of hidden state sequences in LCRF based abbreviation generator.

To implicitly incorporate non-local information, we employ LCRFs [Morency et al. 2007; Petrov and Klein 2008] for abbreviating terms. The LCRF model uses latent variables to capture additional information that may not be expressed by the observable labels. For example, using LCRFs, a possible feature could be "the current character $x_i = \mathtt{X}$, the label $y_i = \mathtt{P}$, and the latent variable $h_i = \mathtt{LV}$." Non-local information can be effectively modeled in LCRFs, and the additional information at the previous position or many of the other positions in the past could be transferred via the latent variables. Figure 3 illustrates an example of hidden state sequences in LCRF-based abbreviation generator.

Using the label set $Y = \{\mathtt{P}, \mathtt{S}\}$, abbreviation generation is formalized as the task of assigning a sequence of labels $\mathbf{y} = y_1, y_2, \ldots, y_m$ for a given sequence of characters $\mathbf{x} = x_1, x_2, \ldots, x_m$ in an expanded form. Each label, y_j, is a member of the possible labels $\mathbb{Y}$. For each sequence, we also assume a sequence of latent variables $\mathbf{h} = h_1, h_2, \ldots, h_m$, which are unobservable in training examples.

The LCRF model is defined as the following [Morency et al. 2007].

$$P(\mathbf{y}|\mathbf{x}, \mathbf{w}) \triangleq \sum_{\mathbf{h}} P(\mathbf{y}|\mathbf{h}, \mathbf{x}, \mathbf{w}) P(\mathbf{h}|\mathbf{x}, \mathbf{w}),$$

where $\mathbf{w}$ represents the parameter vector of the model. To make the training efficient, a restriction is made for the model: for each label, the latent variables associated with it have no intersection with the latent variables from other labels [Morency et al. 2007; Petrov and Klein 2008]. This simplification is also a popular practice in other latent conditional models, including hidden-state conditional random fields (HCRF) [Quattoni et al. 2007]. Each h is a member in a set $\mathbb{H}(y)$ of possible latent variables for the class label y, and $\mathbb{H}(y_j) \cap \mathbb{H}(y_k) = \varnothing$ if $y_j \neq y_k$. $\mathbb{H}$ is defined as the set of all possible latent variables; that is, the union of all $\mathbb{H}(y)$ sets: $\mathbb{H} = \cup_{y \in \mathbb{Y}} \mathbb{H}(y)$. In other words, h can have any value from $\mathbb{H}$, but $P(y|h)$ is zero except for only one of y in $\mathbb{Y}$. The disjoint restriction indicates a discrete simplification of $P(\mathbf{y}|\mathbf{h}, \mathbf{x}, \mathbf{w})$:

$$P(\mathbf{y}|\mathbf{h}, \mathbf{x}, \mathbf{w}) = 1 \quad \textit{if} \quad \mathbf{h} \in \mathbb{H}(y_1) \times \ldots \times \mathbb{H}(y_m)$$
$$P(\mathbf{y}|\mathbf{h}, \mathbf{x}, \mathbf{w}) = 0 \quad \textit{if} \quad \mathbf{h} \notin \mathbb{H}(y_1) \times \ldots \times \mathbb{H}(y_m)$$

where m is the length of the labeling: $m = |\mathbf{y}|$. The formula $\mathbf{h} \in \mathbb{H}(y_1) \times \ldots \times \mathbb{H}(y_m)$ indicates that the latent-labeling $\mathbf{h}$ is a latent-labeling of the labeling y, which can be more formally defined as the following.

$$\mathbf{h} \in \mathbb{H}(y_1) \times \ldots \times \mathbb{H}(y_m) \iff h_j \in \mathbb{H}(y_j),\ j = 1, \ldots, m$$

Since sequences that have any $h_j \notin \mathbb{H}(y_j)$ will by definition have $P(\mathbf{y}|h_j, \mathbf{x}, \mathbf{w}) = 0$, the model can be simplified as the following.

$$P(\mathbf{y}|\mathbf{x}, \mathbf{w}) \triangleq \sum_{\mathbf{h} \in \mathbb{H}(y_1) \times \ldots \times \mathbb{H}(y_m)} P(\mathbf{h}|\mathbf{x}, \mathbf{w}) \tag{5}$$

The item $P(\mathbf{h}|\mathbf{x},\mathbf{w})$ is defined by the usual conditional random field formulation.

$$P(\mathbf{h}|\mathbf{x},\mathbf{w}) = \frac{\exp\left[\mathbf{w}^{\top}\mathbf{f}(\mathbf{h},\mathbf{x})\right]}{\sum_{\mathbf{h}'\in\mathbb{H}\times\ldots\times\mathbb{H}} \exp\left[\mathbf{w}^{\top}\mathbf{f}(\mathbf{h}',\mathbf{x})\right]}, \tag{6}$$

where $\mathbf{f}(\mathbf{h},\mathbf{x})$ is a global feature vector.

Given a training set consisting of n labeled sequences, $(\mathbf{x}_i,\mathbf{y}_i)$, for $i = 1\ldots n$, parameter estimation is performed by optimizing the objective function

$$\begin{aligned} \mathcal{L}(\mathbf{w}) &= \sum_{i=1}^{n} \log P(\mathbf{y}_i|\mathbf{x}_i,\mathbf{w}) - R(\mathbf{w}) \\ &= \sum_{i=1}^{n} \ell(i,\mathbf{w}) - \frac{||\mathbf{w}||^2}{2\sigma^2}. \end{aligned} \tag{7}$$

3.3. Online Training

Training LCRFs is quite challenging. Traditional gradient-based batch training methods, like Limited-memory BFGS (LBFGS) [Nocedal and Wright 1999], are slow on training LCRFs [Petrov and Klein 2008; Sun et al. 2009a]. For example, Sun et al. [2009a] showed that the training of LCRFs is quite costly on biomedical named entity recognition. To accelerate training speed, we try to employ online optimization methods for training LCRFs. First, we review the literature on stochastic gradient descent (SGD).

3.3.1. Stochastic Gradient Descent. The SGD [Bottou and LeCun 2004] uses a small randomly-drawn subset of the training samples to approximate the gradient of an objective function. In this way, one can update the model parameters much more frequently and speed up the convergence. Suppose $\hat{\mathcal{S}}$ is a randomly drawn subset of the full training set $\mathcal{S}$, the stochastic objective function is then given by

$$\mathcal{L}_{stoch}(\mathbf{w},\hat{\mathcal{S}}) = \sum_{i\in\mathcal{S}} \ell(i,\mathbf{w}) - \frac{|\hat{\mathcal{S}}|}{|\mathcal{S}|}\frac{||\mathbf{w}||^2}{2\sigma^2}.$$

The extreme case is a batch size of 1, and it gives the maximum frequency of updates, which we adopt in this work. In this case, $|\hat{\mathcal{S}}| = 1$ and $|\mathcal{S}| = n$ (suppose the full training set contains n samples). In this case, we have

$$\mathcal{L}_{stoch}(\mathbf{w},\hat{\mathcal{S}}) = \ell(i,\mathbf{w}) - \frac{1}{n}\frac{||\mathbf{w}||^2}{2\sigma^2}, \tag{8}$$

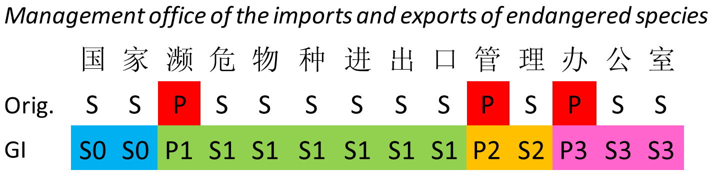
where $\hat{\mathcal{S}} = \{i\}$. The model parameters are updated in the following way.

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \gamma_k \nabla_{\mathbf{w}_k} \mathcal{L}_{stoch}(\mathbf{w},\hat{\mathcal{S}}), \tag{9}$$

where k is the update counter, γ_k is the learning rate.

3.3.2. SGD with Modifications (SGDM). In preliminary experiments, we find the standard SGD training is still slow in this task. The reason of the slowness is from the setting of the learning rates. The setting of learning rates often has a big impact on the convergence speed of the SGD training. A typical choice of learning rate is as follows [Collins et al. 2008].

$$\gamma_k = \frac{\gamma_0}{1+k/n}, \tag{10}$$



Management office of the imports and exports of endangered species

	国	家	濒	危	物	种	进	出	口	管	理	办	公	室
Orig.	S	S	P	S	S	S	S	S	S	P	S	P	S	S
GI	S0	S0	P1	S1	S1	S1	S1	S1	S1	P2	S2	P3	S3	S3

Fig. 4. Comparison of the proposed label encoding method with global information (GI) and the conventional label encoding method.

where γ_0 is a constant. Although this scheduling guarantees theoretical ultimate convergence, the actual convergence speed can be slow in practice [Tsuruoka et al. 2009]. For our task, we want a more effective scheduling of learning rates to achieve faster convergence speed of the SGD training. We tested the following simple exponential decay [Tsuruoka et al. 2009].

$$\gamma_k = \gamma_0 \alpha^{k/n}, \tag{11}$$

where α is constant, with $0 < \alpha < 1$. Our experiments demonstrate that this exponential decay is more effective than the traditional decay (Equation 10), and arrives at a empirical convergence state faster. The main reason is that the exponential decay is more smooth than the traditional decay. The traditional decay schedule drops too quickly at the beginning and too slowly at the end. Although the exponential decay has no guarantee on theoretical ultimate convergence, this is not a problem for practical applications. The reason is that the theoretical ultimate convergence is normally very slow, and the training has to be terminated when the training reaches empirical convergence. Our experiments demonstrate that the exponential decay schedule can reach empirical convergence[2] faster than the traditional decay schedule.

In addition, when the update gradients are sparse and the overall feature dimension is high, it will be wasteful to use the standard SGD updates, because at each update we need to apply regularization to all features, including those features that are not used in the current training sample. To make training faster, we use a heuristic lazy regularization schedule for the SGD.

3.4. Label Encoding with Global Information

We can design the labels such that they explicitly incorporate non-local information. In this approach, the label y_i at position i attaches the information of the abbreviation length generated by its previous labels, $y_1, y_2, \ldots, y_{i-1}$. Figure 4 shows an example of a Chinese abbreviation. In this encoding, a label not only contains the *produce or skip* information, but also the abbreviation-length information, that is, the label includes the number of all P labels preceding the current position. We refer to this method as *label encoding with global information* (hereinafter, *GI*).

Note that the model-complexity is increased only by the increase in the number of labels. Since the length of the abbreviations is usually quite short (less than 5 for Chinese abbreviations and less than 10 for English abbreviations), the model is still tractable even when using the GI encoding.

The implicit (LCRF) and explicit (GI) solutions address the same issue concerning the incorporation of non-local information, and there are advantages to combining these two solutions. Therefore, we will combine the implicit and explicit solutions by

[2] In practical applications, *empirical convergence* is employed to evaluate convergence state. Empirical convergence is typically defined such that the relative improvement of objective function is less than a real-valued threshold.

Table I. Language-Independent Features (#1 to #3), Chinese-Specific Features (#4 Through #7), and English-Specific Features (#8 Through #11)

#1	The input char. x_{i-1} and x_i
#2	Whether x_j is a numeral, for $j = (i-3)\dots i$
#3	The char. bigrams starting at $(i-2)\dots i$
#4	The *Pinyin* of char. x_{i-1} and x_i
#5	The *Pinyin* bigrams starting at $(i-2)\dots i$
#6	Whether $x_j = x_{j+1}$, for $j = (i-2)\dots i$
#7	Whether $x_j = x_{j+2}$, for $j = (i-3)\dots i$
#8	Whether x_j is uppercase, for $j = (i-3)\dots i$
#9	Whether x_j is lowercase, for $j = (i-3)\dots i$
#10	The char. 3-grams starting at $(i-3)\dots i$
#11	The char. 4-grams starting at $(i-4)\dots i$

employing the GI encoding in the LCRF (LCRF+GI). The effects of this combination will be demonstrated through experiments.

3.5. Feature Design

Next, we design two types of features: language-independent features and language-specific features. Language-independent features can be used for abbreviating terms in English and Chinese. We use the features from #1 to #3 listed in Table I.

Feature templates #4 to #7 in Table I are used for Chinese abbreviations. Templates #4 and #5 express the *Pinyin* reading of the characters, which represents a Romanization of the sound. Templates #6 and #7 are designed to detect character duplication, because identical characters will normally be skipped in the abbreviation process. On the other hand, such duplication detection features are not so useful for English abbreviations.

Feature templates #8–#11 are designed for English abbreviations. Features #8 and #9 encode the orthographic information of expanded forms. Features #10 and #11 represent a contextual n-gram with a large window size. Since the number of letters in Chinese (more than $10K$ characters) is much larger than the number of letters in English (26 letters), in order to avoid a possible overfitting problem, we did not apply these feature templates to Chinese abbreviations.

4. EXPERIMENTS

Chinese newswire data. For Chinese newswire abbreviation generation, we used the corpus of Sun et al. [2008a], which contains 2,914 abbreviation definitions for training, and 729 pairs for testing. This corpus consists primarily of noun phrases (38%), organization names (32%), and verb phrases (21%).

English biomedical data. For English biomedical abbreviation generation, we evaluated on the corpus of Tsuruoka et al. [2005]. This corpus contains 1,200 aligned pairs extracted from MEDLINE biomedical abstracts published in 2001. For both tasks, we converted the aligned pairs of the corpora into labeled full forms and used the labeled full forms as the training/testing data.

4.1. Experimental Settings

We employ the feature templates defined in Section 3.5, taking into account these 81,827 features for the Chinese abbreviation generation task, and the 50,149 features for the English abbreviation generation task. We use three latent variables for each

Table II. Results of Chinese Abbreviation Generation. T1A, T2A, and T3A Represent Top-1, Top-2, and Top-3 Accuracy, Respectively. The System Marked with the * Symbol is the Recommended System

Model	T1A (%)	T2A (%)	T3A (%)	Train-Time
Heu (Sun08)	41.6	N/A	N/A	N/A
HMM (Sun08)	46.1	N/A	N/A	N/A
SVM (Sun08)	62.7	80.4	87.7	1.3 h
CRF	64.5	81.1	88.7	0.2 h
CRF+GI	66.8	82.5	90.0	0.5 h
LCRF	67.6	83.8	91.3	0.4 h
LCRF+GI (*)	**72.3**	**87.6**	**94.9**	1.1 h

original label in LCRFs.[3] We will show the experimental results on varying the number of latent variables.

For numerical optimization, we performed a gradient descent with the Limited-Memory BFGS (LBFGS) optimization technique [Nocedal and Wright 1999]. LBFGS is a second-order Quasi-Newton method that numerically estimates the curvature from previous gradients and updates. With no requirement on specialized Hessian approximation, LBFGS can handle large-scale problems efficiently. Since the objective function of the LCRF model is non-convex, different parameter initializations normally bring different optimization results. Therefore, to approach closer to the global optimal point, it is recommended to perform multiple experiments on LCRFs with random initialization and then select a good start point.

Note that, for the label encoding with global information, many label transitions (e.g., $\mathtt{P}_2\mathtt{S}_3$) are actually impossible: the label transitions are strictly constrained, that is, $y_i y_{i+1} \in \{\mathtt{P}_\mathtt{j}\mathtt{S}_\mathtt{j}, \mathtt{P}_\mathtt{j}\mathtt{P}_\mathtt{j+1}, \mathtt{S}_\mathtt{j}\mathtt{P}_\mathtt{j+1}, \mathtt{S}_\mathtt{j}\mathtt{S}_\mathtt{j}\}$. These constraints on the model forward-backward lattice are enforced by giving appropriate features a weight of $-\infty$, thereby forcing all forbidden labelings to have zero probability. Sha and Pereira [2003] originally proposed this concept of implementing transition restrictions.

The evaluation metrics used in the abbreviation generation are *exact-match accuracy* (hereinafter *accuracy*), including top-1 accuracy, top-2 accuracy, and top-3 accuracy. The top-N accuracy represents the percentage of correct abbreviations that are covered, if we take the top N candidates from the ranked labelings of an abbreviation generator.

4.2. Results on Chinese Newswire Data

First, we present the results of the Chinese abbreviation generation task, as listed in Table II. To evaluate the impact of using latent variables, we chose a representative non-latent model, the CRF model, as a baseline. We compared the performance of the LCRF with the CRFs and other baseline systems, including the heuristic system (Heu), the HMM model, and the SVM model described in *Sun08*, that is, Sun et al. [2008a]. The heuristic method is a simple rule that produces the initial character of each word to generate the corresponding abbreviation. The SVM method described by Sun et al. [2008a] is formalized as a regression problem, in which the abbreviation candidates are scored and ranked.

The results revealed that the latent variable model considerably improved the performance over the CRF model. All of its top-1, top-2, and top-3 accuracies were consistently better than those of the CRF model. Therefore, this demonstrated the effectiveness of using the latent variables in Chinese abbreviation generation. The

[3] This is a correction of the number of latent variables discussed in Sun et al. [2009b].

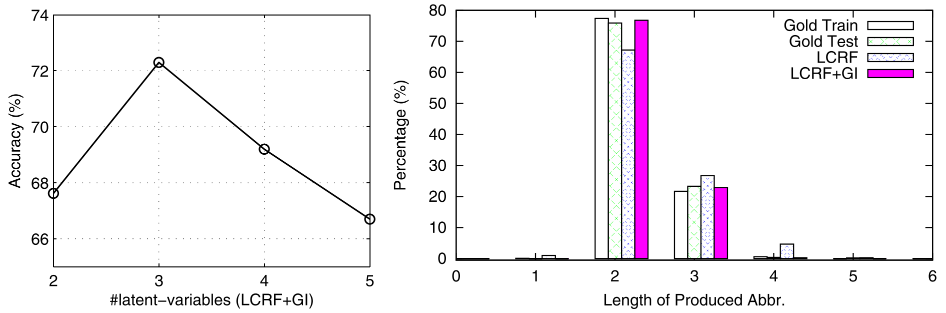


Fig. 5. (Left) Results on varying the number of latent variables. Since LCRF+GI performed well, the experiments are based on LCRF+GI. (Right) Percentage distribution of Chinese abbreviations/Viterbi-labelings grouped by length.

Table III. Results of English Abbreviation Generation

Model	T1A (%)	T2A (%)	T3A (%)	Train-Time
CRF	55.8	65.1	70.8	0.3 h
CRF+GI	52.7	63.2	68.7	1.3 h
CRF+GIB	56.8	66.1	71.7	1.3 h
LCRF	57.6	67.4	73.4	0.6 h
LCRF+GI	53.6	63.2	69.2	2.5 h
LCRF+GIB (*)	**58.3**	N/A	N/A	3.0 h

experimental results on varying the number of latent variables are shown in Figure 5. Since LCRF+GI performed well, the experiments are based on LCRF+GI.

As the case for the two alternative approaches for incorporating non-local information, the latent variable method and the label encoding method competed with one another (see LCRF vs. CRF+GI). The results showed that the latent variable method outperformed the GI encoding method by +0.8% on the top-1 accuracy. The reason for this could be that the label encoding approach is a solution without the adaptivity on different instances. We will present a detailed discussion comparing LCRF and CRF+GI for the English abbreviation generation task in the next subsection, where the difference is more significant.

In contrast, to a larger extent, the results demonstrate that these two alternative approaches are complementary. Using the GI encoding further improved the performance of the LCRF, with +4.7% on top-1 accuracy. We found that major improvements were achieved through the more exact control of the output length. To perform a detailed analysis, we collected the statistics of the length distribution (see Figure 5) and found that the GI encoding improved the abbreviation length distribution of the LCRF.

The proposed method, the latent variable model with GI encoding, is 9.6% better with respect to the top-1 accuracy compared to the best system on this corpus, namely, the SVM regression method. Furthermore, the top-3 accuracy of the latent variable model with GI encoding is as high as 94.9%, which is quite encouraging for practical usage.

4.3. Results on English Biomedical Data

In the English abbreviation generation task, we randomly selected 1,481 instances from the generation corpus for training, and 370 instances for testing. Table III shows the experimental results. We compared the performance of the LCRF with the performance of the CRFs. Whereas the use of the latent variables still improves the generation performance, using the GI encoding hurted the performance in this task. In

```
    somatosensory evoked potentials
(a) P1P2          P3     P4       P5 SMEPS
(b) P             P      P        P  SEPS
(a): CRF+GI with p=0.001    [Wrong]
(b): LCRF  with p=0.191    [Correct]
```

Fig. 6. A result of “CRF+GI vs. LCRF”. For simplicity, the S labels are masked.

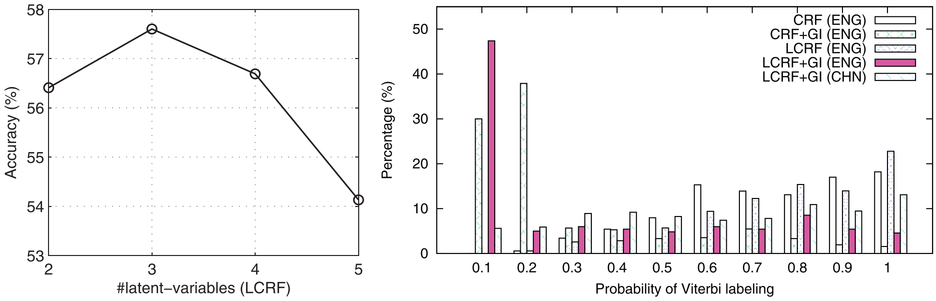


Fig. 7. (Left) Results on varying the number of latent variables. Since LCRF performed well in this dataset, the experiments are based on LCRF. (Right) For various models, the probability distributions of the produced abbreviations on the test data of the English abbreviation generation task.

```
           mitomycin C
    LCRF   P          P   MC   [Wrong]
 LCRF+GI   P1   P2    P3  MMC  [Correct]
```

Fig. 8. Example of abbreviations composed of non-initials generated by the LCRF and the LCRF+GI.

comparing the implicit and explicit solutions for incorporating non-local information, we can see that the implicit approach with LCRFs performs much better than the explicit approach with the GI encoding. An example is shown in Figure 6. The CRF+GI produced a Viterbi labeling with a low probability, which is an incorrect abbreviation. The LCRF produced the correct labeling. The experimental results on varying the number of latent variables are shown in Figure 7. Since LCRF performed well in this dataset, the experiments are based on LCRF.

To perform a systematic analysis of the superior-performance of LCRF compare to CRF+GI, we collected the probability distributions (see Figure 7) of the Viterbi labelings from these models (see “LCRF vs. CRF+GI”). The curves suggest that the data sparseness problem could be the reason for the differences in performance. A large percentage (37.9%) of the Viterbi labelings from the CRF+GI (ENG) have very small probability values ($p < 0.1$). For the LCRF (ENG), there were only a few (0.5%) Viterbi labelings with small probabilities. Since English abbreviations are often longer than Chinese abbreviations ($length < 10$ in English, whereas $length < 5$ in Chinese), using the GI encoding resulted in a larger label set in English. Hence, the features become more sparse than in the Chinese case. In addition, the training data of the English task is much smaller than the Chinese task, which could make the models more sensitive to data sparseness. Therefore, a significant number of features could have been inadequately trained, resulting in Viterbi labelings with low probabilities. For the latent variable approach, its curve demonstrates that it did not cause a severe data sparseness problem.

The aforementioned analysis also explains the poor performance of the LCRF+GI. However, the LCRF+GI can actually produce correct abbreviations with high probabilities in some difficult instances. In Figure 8, the LCRF produced an incorrect labeling

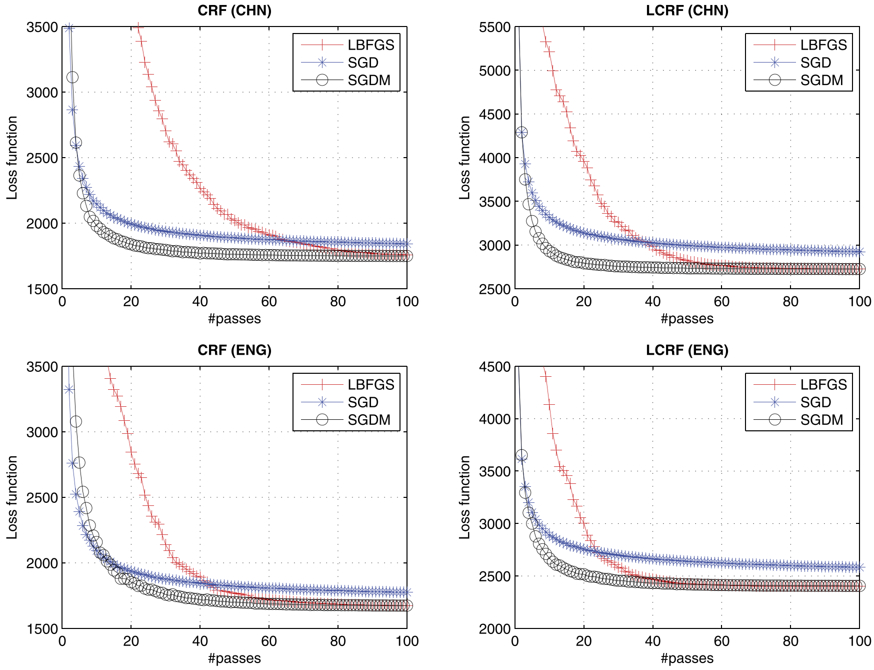


Fig. 9. Log-likelihood curves of different training methods on different models and different data sets. CHN stands for Chinese. ENG stands for English.

for the difficult long form, whereas the LCRF+GI produced the correct labeling containing non-initials.

Hence, we present a simple voting method to better combine the latent variable approach with the GI encoding method. We refer to this new combination as GI encoding with *back-off* (hereinafter *GIB*): when the abbreviation generated by the LCRF+GI has a confident probability ($p > 0.3$ in the present case), the LCRF+GI then outputs it. Otherwise, the system backs-offs to the parameters trained without the GI encoding, that is, the LCRFs.

The results in Table III demonstrate that the LCRF+GIB model significantly outperformed the other models because the LCRF+GI model improved the performance in some difficult instances. The LCRF+GIB model was robust even when the data sparseness problem was severe.

To strictly compare with the previous results in Tsuruoka et al. [2005], following Tsuruoka et al. [2005], we performed a five-fold cross-validation on the corpus. Our LCRF method achieved the T1A of 57.5%, which is better than the T1A in Tsuruoka et al. [2005]. In Tsuruoka et al. [2005], the highest T1A report is 55.2% by using a maximum entropy Markov model (MEMM).

5. ACCELERATED TRAINING

Here, we perform experiments on optimizing objective functions of CRFs and LCRFs on different data sets. We compare our SGD with modifications (SGDM) with the popular LBFGS batch training method and the standard SGD training method. For the SGDM method, we need to tune the value of γ_0 and α. The tuning of γ_0 and α was performed in such a way that follows previous work [Tsuruoka et al. 2009].

We tested the three training methods on CRF/LCRF and different data sets, with English biomedical data and Chinese newswire data. The curves of the objective

Table IV. Training Time and Iterations of Different Training Methods on Different Models and Different Data Sets

Experiments are based on a 2.13GHz CPU. Since the LCRF+GIB is a combination of LCRF and LCRF+GI, the training time of LCRF+GIB can be directly derived from LCRF and LCRF+GI, that is, the summation of LCRF and LCRF+GI.

Models	Opt. Methods	CHN Train-Time (s)	#Passes	ENG Train-Time (s)	#Passes
LCRF	LBFGS	784	90	530	90
	SGDM	401	40	318	40
LCRF+GI	LBFGS	3846	100	14914	100
	SGDM	1623	40	8170	50
LCRF+GIB	LBFGS	4630	90/100	15444	90/100
	SGDM	2024	40/40	8488	40/50

functions by varying training iterations are shown in Figure 9. As we can see, in all cases compared with the LBFGS batch training method, the SGDM method achieved the same level of objective values on convergence. Most importantly, in most of the cases, the SGDM method converges much faster than the LBFGS batch training method. The detailed training-time and iterations of SGDM and LBFGS are shown in Table IV[4].

It is noteworthy that the weights produced by the SGDM and the LBFGS training are very similar when the two methods optimized the objective function to the same optimum. Hence, in spite of the significant acceleration of training speed, abbreviation generation accuracies are almost the same when we use the SGDM training instead of the LBFGS training. We do not repeat the accuracy results here.

6. CONCLUSIONS AND FUTURE RESEARCH

We have presented the latent conditional random fields and the GI encoding to incorporate non-local information in abbreviating terms. We have shown that each of those two methods can be used to model non-local information. On the other hand, more importantly, we showed that the two approaches were complementary to each other. By combining the two approaches, we were able to achieve state-of-the-art performance in abbreviation generation in the same model, across different languages, and with a simple feature set. Not surprisingly, experimental results showed that learning non-local information may slow down the training speed. To solve this problem, we further presented an online training method, which can arrive at the same optimum with accelerated training speed.

As discussed earlier herein, the training data is relatively small. Since there are numerous unlabeled full forms on the web, it is possible to use a semi-supervised approach in order to make use of such raw data. This is an area for future research.

ACKNOWLEDGMENTS

We thank all of the reviewers for their insightful comments which improved this article.

REFERENCES

Adar, E. 2004. SaRAD: A simple and robust abbreviation dictionary. *Bioinform. 20*, 4, 527–533.

Ao, H. and Takagi, T. 2005. ALICE: An algorithm to extract abbreviations from MEDLINE. *J. Amer. Med. Inform. Assoc. 12*, 5, 576–586.

Barrett, J. A. and Grems, M. 1960. Abbreviating words systematically. *Comm. ACM 3*, 5, 323–324.

[4] Since computing platforms like CPUs and algorithm implementations in this table are different from previous ones, the results of time are only for comparisons within this table.

Bottou, L. and LeCun, Y. 2004. Large scale online learning. In *Advances in Neural Information Processing Systems 16*, S. Thrun, L. Saul, and B. Scholkopf Eds., MIT Press, Cambridge, MA.

Bourne, C. P. and Ford, D. F. 1961. A study of methods for systematically abbreviating English words and names. *J. ACM 8*, 4, 538–552.

Chang, J. T. and Schütze, H. 2006. Abbreviations in biomedical text. In *Text Mining for Biology and Biomedicine*, S. Ananiadou and J. McNaught Eds., Artech House, Inc., 99–119.

Collins, M., Globerson, A., Koo, T., Carreras, X., and Bartlett, P. L. 2008. Exponentiated gradient algorithms for conditional random fields and max-margin Markov networks. *J. Mach. Learn. Res. 9*, 1775–1822.

Frantzi, K. T. and Ananiadou, S. 1999. The C-value/NC-value domain independent method for multi-word term extraction. *J. Natural Lang. Proc. 6*, 3, 145–179.

HaCohen-Kerner, Y., Kass, A., and Peretz, A. 2008. Combined one sense disambiguation of abbreviations. In *Proceedings of the Human Language Technology Conference* (Short Papers) *(HLT'08)*. 61–64.

Hisamitsu, T. and Niwa, Y. 2001. Extracting useful terms from parenthetical expression by combining simple rules and statistical measures: A comparative evaluation of bigram statistics. In *Recent Advances in Computational Terminology*, D. Bourigault, C. Jacquemin, and M.-C. L'Homme Eds., John Benjamins, 209–224.

Kuo, C.-J., Ling, M. H., Lin, K.-T., and Hsu, C.-N. 2009. BIOADI: A machine learning approach to identifying abbreviations and definitions in biological literature. *BMC Bioinform. 10*, Suppl. 15, S7.

Lafferty, J., McCallum, A., and Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML'01)*. 282–289.

Liu, H. and Friedman, C. 2003. Mining terminological knowledge in large biomedical corpora. In *Proceedings of the 8th Pacific Symposium on Biocomputing (PSB'03)*. 415–426.

Morency, L.-P., Quattoni, A., and Darrell, T. 2007. Latent-dynamic discriminative models for continuous gesture recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR'07)*. 1–8.

Nadeau, D. and Turney, P. D. 2005. A supervised learning approach to acronym identification. In *Proceedings of the 8th Canadian Conference on Artificial Intelligence (AI'05)*.

Nocedal, J. and Wright, S. J. 1999. *Numerical Optimization*. Springer.

Okazaki, N. and Ananiadou, S. 2006. Building an abbreviation dictionary using a term recognition approach. *Bioinform. 22*, 24, 3089–3095.

Okazaki, N., Ananiadou, S., and Tsujii, J. 2008. A discriminative alignment model for abbreviation recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*. 657–664.

Pakhomov, S. 2002. Semi-supervised maximum entropy based approach to acronym and abbreviation normalization in medical texts. In *Proceedings of the Association of Computer Linguistics (ACL'02)*. 160–167.

Park, Y. and Byrd, R. J. 2001. Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the Joint Meeting of the Conference on Empirical Methods in Natural Language Processing (EMNLP'01)*. 126–133.

Petrov, S. and Klein, D. 2008. Discriminative log-linear grammars with latent variables. In *Proceedings of the Conference on Advances in Neural Information Processing Systems 20 (NIPS'08)*. 1153–1160.

Quattoni, A., Wang, S., Morency, L.-P., Collins, M., and Darrell, T. 2007. Hidden conditional random fields. *IEEE Trans. Pattern Anal. Mach. Intell. 29*, 10, 1848–1852.

Schwartz, A. S. and Hearst, M. A. 2003. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Proceedings of the 8th Pacific Symposium on Biocomputing (PSB'03)*. 451–462.

Sha, F. and Pereira, F. 2003. Shallow parsing with conditional random fields. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (NAACL'03)*. 134–141.

Sun, X. and Tsujii, J. 2009. Sequential labeling with latent variables: An exact inference algorithm and its efficient approximation. In *Proceedings of the Conference on the European Chapter of the Association for Computational Linguistics (EACL'09)*. 772–780.

Sun, X., Wang, H., and Wang, B. 2008a. Predicting Chinese abbreviations from definitions: An empirical learning approach using support vector regression. *J. Comp. Sci. Tech. 23*, 4, 602–611.

Sun, X., Morency, L.-P., Okanohara, D., and Tsujii, J. 2008b. Modeling latent-dynamic in shallow parsing: A latent conditional model with improved inference. In *Proceedings of the International Conference on Computer Linguistics (COLING'08)*. 841–848.

Sun, X., Matsuzaki, T., Okanohara, D., and Tsujii, J. 2009a. Latent variable perceptron algorithm for structured classification. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*. 1236–1242.

Sun, X., Okazaki, N., and Tsujii, J. 2009b. Robust approach to abbreviating terms: A discriminative latent variable model with global information. In *Proceedings of the Association of Computer Linguistics (ACL'09)*. 905–913.

Taghva, K. and Gilbreth, J. 1999. Recognizing acronyms and their definitions. *Int. J. Doc. Anal. Recog. 1*, 4, 191–198.

Torii, M., Zhi Hu, Z., Song, M., Wu, C. H., and Liu, H. 2007. A comparison study on algorithms of detecting long forms for short forms in biomedical text. *BMC Bioinform. 8*, Suppl 9, S5.

Tsuruoka, Y., Ananiadou, S., and Tsujii, J. 2005. A machine learning approach to acronym generation. In *Proceedings of the International Conference on Intelligent Systems in Molecular Biology (ISMB'05)*. 25–31.

Tsuruoka, Y., Tsujii, J., and Ananiadou, S. 2009. Stochastic gradient descent training for l1-regularized log-linear models with cumulative penalty. In *Proceedings of the Association for Computer Linguistics (ACL'09)*. 477–485.

Wakaki, H., Fujii, H., Suzuki, M., Fukui, M., and Sumita, K. 2009. Abbreviation generation for Japanese multi-word expressions. In *Proceedings of the Workshop on Multiword Expressions: Identification, Interpretation, Disambiguation and Applications (MWE'09)*. 63–70.

Wren, J. D. and Garner, H. R. 2002. Heuristics for identification of acronym-definition patterns within text: Towards an automated construction of comprehensive acronym-definition dictionaries. *Meth. Inf. Med. 41*, 5, 426–434.

Xu, J. and Huang, Y. 2006. Using SVM to extract acronyms from text. *J. Soft Comput. - A Fusion Found. Method. Appli. 11*, 4.

Yamamoto, Y., Yamaguchi, A., Bono, H., and Takagi, T. 2011. Allie: A database and a search service of abbreviations and long forms. *J. Biol. Datab. Curation 2011*, bar013.

Yang, D., Pan, Y.-C., and Furui, S. 2009. Vocabulary expansion through automatic abbreviation generation for Chinese voice search. In *Proceedings of the European Conference on Speech Communication and Technology (INTERSPEECH'09)*. 728–731.

Yarowsky, D. 1993. One sense per collocation. In *Proceedings of the Workshop on Human Language Technology (HLT'93)*. 266–271.

Yeganova, L., Comeau, D. C., and Wilbur, W. J. 2010. Identifying abbreviation definitions—Machine learning with naturally labeled data. In *Proceedings of the 9th International Conference on Machine Learning and Application (ICMLA'10)*. 499–505.

Yu, H., Hripcsak, G., and Friedman, C. 2002. Mapping abbreviations to full forms in biomedical articles. *J. Amer. Med. Inform. Assoc. 9*, 3, 262–272.

Yu, H., Kim, W., Hatzivassiloglou, V., and Wilbur, J. 2006. A large scale, corpus-based approach for automatically disambiguating biomedical abbreviations. *ACM Trans. Inf. Sys. 24*, 3, 380–404.

Zhou, W., Torvik, V. I., and Smalheiser, N. R. 2006. ADAM: Another database of abbreviations in MEDLINE. *Bioinform. 22*, 22, 2813–2818.

Received December 2011; revised February 2012; accepted May 2012