Chinese Named Entity Abbreviation Generation Using First-Order Logic

Huan Chen, Qi Zhang, Jin Qian, Xuanjing Huang
School of Computer Science
Fudan University
Shanghai, P.R. China
{12210240054, qz, 12110240030, xjhuang}@fudan.edu.cn

Abstract

Normalizing named entity abbreviations to their standard forms is an important preprocessing task for question answering, entity retrieval, event detection, microblog processing, and many other applications. Along with the quick expansion of microblogs, this task has received more and more attentions in recent years. In this paper, we propose a novel entity abbreviation generation method using first-order logic to model long distance constraints. In order to reduce the human effort of manual annotating corpus, we also introduce an automatically training data construction method with simple strategies. Experimental results demonstrate that the proposed method achieves better performance than state-of-the-art approaches.

1 Introduction

Twitter and other social media services have received considerable attentions in recent years. Users provide hundreds of millions microblogs through them everyday. The informative data has been relied on by many applications, such as sentiment analysis (Jiang et al., 2011; Meng et al., 2012), event detection (Sakaki et al., 2010; Lin et al., 2010), stock market predication (Bollen et al., 2011), and so on. However, due to the constraint on the length of characters, abbreviations frequently occur in microblogs. According to a statistic, approximately 20% of sentences in news articles have abbreviated words (Chang and Lai, 2004). The frequency of abbreviation has become even more popular along with the rapid increment of user generated contents. Without pre-normalizing these abbreviations, most of the natural language processing systems may heavily suffer from them.

The goal of entity abbreviation generation is to produce abbreviated equivalents of the original entities. Table 1 shows several examples of entities and their corresponding abbreviations. A few of approaches have been done on this task. Li and Yarowsky (Li and Yarowsky, 2008b) introduced an unsupervised method used to extract phrases and their abbreviation pair using parallel dataset and monolingual corpora. Xie et al. (2011) proposed to use weighted bipartite graph to extract definition and corresponding abbreviation pairs from anchor texts. Since these methods rely heavily on lexical/phonetic similarity, substitution of characters and portion may not be correctly identified through them. Yang et al. (2009) studied the Chinese entity name abbreviation problem. They formulated the abbreviation task as a sequence labeling problem and used the conditional random fields (CRFs) to model it. However the long distance and global constraint can not be easily modeled thorough CRFs.

Entity	Abbr.
北京大学 (Peking University)	北大
中国石油天然气集团公司 (China National Petroleum Corporation)	中石油
中国国际航空公司 (Air China)	国航

Table 1: Abbreviation examples

To overcome these limitations, in this paper, we propose a novel entity abbreviation generation method, which combines first-order logic and rich linguistic features. To the best of our knowledge, our approach is the first work of using first-order logic for this entity abbreviation. Abbreviation generation is converted to character deletion and

International Joint Conference on Natural Language Processing, pages 320–328,
Nagoya, Japan, 14-18 October 2013.

keep operations which are modeled by logic formula. Linguistic features and relations between different operations are represented by local and global logic formulas respectively. Markov Logic Networks (MLN) (Richardson and Domingos, 2006) is adopted for learning and predication. To reduce the human effort in constructing the training data, we collect standard forms of entities from online encyclopedia and introduce a few of simple patterns to extract abbreviations from documents and search engine snippets with high precision as training data. Experimental results show that the proposed methods achieve better performance than state-of-the-art methods and can efficiently process large volumes of data.

The remainder of the paper is organized as follows: In section 2, we review a number of related works and the state-of-the-art approaches in related areas. Section 3 presents the proposed method. Experimental results in test collections and analyses are shown in section 4. Section 5 concludes this paper.

2 Related Work

The proposed approach builds on contributions from two research communities: text normalization, and Markov Logic Networks. In the following of this section, we give brief description of previous works on these areas.

2.1 Text Normalization

Named entity normalization, abbreviation generation, and lexical normalization are related to this task. These problems have been recognized as important problems for various languages. Since different languages have their own peculiarities, many approaches have been proposed to handle variants of words (Aw et al., 2006; Liu et al., 2012; Han et al., 2012) and named entities (Yang et al., 2009; Xie et al., 2011; Li and Yarowsky, 2008b).

Chang and Teng (2006) introduced an HMM-based single character recovery model to extract character level abbreviation pairs for textual corpus. Okazaki et al. (2008) also used discriminative approach for this task. They formalized the abbreviation recognition task as a binary classification problem and used Support Vector Machines to model it. Yang et al. (2012) also treated the abbreviation generation problem as a labeling task and used Conditional Random Fields (CRFs) to do it. They also proposed to re-rank candidates by a length model and web information.

Li and Yarowsky (2008b) proposed an unsupervised method extracting the relation between a full-form phrase and its abbreviation from monolingual corpora. They used data co-occurrence intuition to identify relations between abbreviation and full names. They also improved a statistical machine translation by incorporating the extracted relations into the baseline translation system. Based on the data co-occurrence phenomena, they introduced a bootstrapping procedure to identify formal-informal relations informal phrases in web corpora (Li and Yarowsky, 2008a) . They used search engine to extract contextual instances of the given an informal phrase, and ranked the candidate relation pairs using conditional log-linear model. Xie et al. (2011) proposed to extract Chinese abbreviations and their corresponding definitions based on anchor texts. They constructed a weighted URL-AnchorText bipartite graph from anchor texts and applied co-frequency based measures to quantify the relatedness between two anchor texts.

For lexical normalisation, Aw et al. (2006) treated the lexical normalisation problem as a translation problem from the informal language to the formal English language and adapted a phrase-based method to do it. Han and Baldwin (2011) proposed a supervised method to detect ill-formed words and used morphophonemic similarity to generate correction candidates. Liu et al. (2012) proposed to use a broad coverage lexical normalization method consisting three key components enhanced letter transformation, visual priming, and string/phonetic similarity. Han et al. (2012) introduced a dictionary based method and an automatic normalisation-dictionary construction method. They assumed that lexical variants and their standard forms occur in similar contexts.

In this paper, we focused on named entity abbreviation generation problem and treated the problem as a labeling task. Due to the flexibilities of Markov Logic Networks on capturing local and global linguistic feature, we adopted it to model the supervised classification procedure. To reduce the human effort in constructing training data, we also introduced a sample rule based method to find relations between standard forms and abbreviations.

Predicates about characters in the entity	
character(i,c)	The ith character is c.
isNumber(i)	The ith character is a number.
Predicates about words in the entity	
word(j,w)	The jth word is w.
isCity(j)	The jth word is a city name.
lastWord(j)	The jth word is the last word.
sufCorp(j)	The jth word belongs the set of common suffixes of corporation.
sufSchool(j)	The jth word belongs the set of common suffixes of school.
sufOrg(j)	The jth word belongs the set of common suffixes of organizations.
sufGov(j)	The jth word belongs the set of common suffixes of government agencies.
idf(j,v)	The inverse document frequency of jth word is v.
Predicates about entire entity	
entityType(t)	The type of the entity is t.
lenChar(n)	The total number of characters is n.
lenWord(n)	The total number of words is n.
Predicates about relations between characters and words	
cwMap(i,j)	The ith character belongs to jth word.
cwPosition(i,j)	The ith character of the entity is the jth character in the corresponding word.

Table 2: Descriptions of observed predicates.

2.2 Markov Logic Networks

Richardson and Domingos (2006) proposed Markov Logic Networks (MLN), which combines first-order logic and probabilistic graphical models. MLN framework has been adopted for several natural language processing tasks and achieved a certain level of success (Singla and Domingos, 2006; Riedel and Meza-Ruiz, 2008; Yoshikawa et al., 2009; Andrzejewski et al., 2011; Jiang et al., 2012; Huang et al., 2012).

Singla and Domingos (2006) modeled the entity resolution problem with MLN. They demonstrated the capability of MLN to seamlessly combine a number of previous approaches. Poon and Domingos (2008) proposed to use MLN for joint unsupervised coreference resolution. Yoshikawa et al. (2009) proposed to use Markov logic to incorporate both local features and global constraints that hold between temporal relations. Andrzejewski et al. (2011) introduced a framework for incorporating general domain knowledge, which is represented by First-Order Logic (FOL) rules, into LDA inference to produce topics shaped by both the data and the rules.

3 The Proposed Approach

In this section, firstly, we briefly describe the Markov Logic Networks framework. Then, we present the first-order logic formulas including local formulas and global formulas we used in this work.

3.1 Markov Logic Networks

A MLN consists of a set of logic formulas that describe first-order knowledge base. Each formula consists of a set of first-order predicates, logical connectors and variables. Different with first-order logic, these hard logic formulas are softened and can be violated with some penalty (the weight of formula) in MLN.

We use $\mathcal{M}$ to represent a MLN and $\{(\phi_i, w_i)\}$ to represent formula ϕ_i and its weight w_i. These weighted formulas define a probability distribution over sets of possible worlds. Let y denote a possible world, the $p(y)$ is defined as follows (Richardson and Domingos, 2006):

$$p(y) = \frac{1}{Z} exp\left(\sum_{(\phi_i, w_i)\in\mathcal{M}} w_i \sum_{c\in C^{n_{\phi_i}}} f_c^{\phi_i}(y)\right),$$

where each c is a binding of free variable in ϕ_i to constraints; $f_c^{\phi_i}(y)$ is a binary feature function that

returns 1 if the true value is obtained in the ground formula we get by replacing the free variables in ϕ_i with the constants in c under the given possible world y, and 0 otherwise; $C^{n_{\phi_i}}$ is all possible bindings of variables to constants, and Z is a normalization constant.

Many methods have been proposed to learn the weights of MLNs using both generative and discriminative approaches (Richardson and Domingos, 2006; Singla and Domingos, 2006). There are also several MLNs learning packages available online such as thebeast[1], Tuffy[2], PyMLNs[3], Alchemy[4], and so on.

3.2 MLN for Abbreviation Generation

In this work, we convert the abbreviation generation problem as a labeling task for every characters in entities. Predicate $drop(i)$ indicates that the character at position i is omitted in the abbreviation. Previous works (Chang and Lai, 2004; Yang et al., 2009) show that Chinese named entities can be further segmented into words. Words also provide important information for abbreviation generation. Hence, in this work, we also segment named entities into words and propose an observed predict to connect words and characters.

3.2.1 Local Formulas

The local formulas relate one or more observed predicates to exactly one hidden predicate. In this work, we define a list of observed predicates to describe the properties of individual characters. Table 2 shows the list. For this task, there is only one hidden predicate $drop$.

Table 3 lists the local formulas used in this work. The "+" notation in the formulas indicates that the each constant of the logic variable should be weighted separately. For example, formula $character(2,二) \wedge isNumber(2) \Rightarrow drop(2)$ and $character(2,十) \wedge isNumber(2) \Rightarrow drop(2)$ may have different weights as inferred by formula $character(i, c+) \wedge isNumber(i) \Rightarrow drop(i)$.

Three kinds of local formulas are introduced in this work. Lexical features are used to capture the context information based on both character and word level information. Distance and position features are helpful in determining which parts of a entity may be removed. Hence, we also incorporate position information of word into local formulas. For example, "大学(University)" is usually omitted when it is at the end of the entity. In practice, abbreviations of some kinds of entities can be generated through several strategies. So we introduce several local formulas to handle a group of related entities with similar suffix.

3.2.2 Global Formulas

Global formulas are designed to handle deletion of multiple characters. Since in this work, we only have one hidden predicate, $drop$, the global formulas incorporate correlations among different ground atoms of the $drop$ predicate.

We propose to use global formulas to force the abbreviations to contain at least 2 characters and to make sure that at least one character is deleted. The following formulas are implemented:

$$|character(i, c) \wedge drop(i)| \text{ all } i| \geqslant 1$$
$$|character(i, c) \wedge \neg drop(i)| \text{ all } i| \geqslant 2$$

Another constraint is that for the characters in some particular words should by dropped or kept simultaneously. So we add two formulas to model this:

$$\begin{aligned} &character(i, c1) \wedge cwMap(i, j) \wedge drop(i) \wedge \\ &character(i+1, c2) \wedge cwMap(i+1, j) \\ &\Rightarrow drop(i+1) \end{aligned}$$

$$\begin{aligned} &character(i, c1) \wedge cwMap(i, j) \wedge \neg drop(i) \wedge \\ &character(i+1, c2) \wedge cwMap(i+1, j) \\ &\Rightarrow \neg drop(i+1) \end{aligned}$$

4 Experiments

In this section, we first describe the dataset construction method, evaluation metrics, and experimental configurations. We then describe the evaluation results and analysis.

4.1 Data Set

For training and evaluating the performance the proposed method, we need a large number of abbreviation and corresponding standard form pairs. However, manually labeling is a laborious and time consuming work. To reduce human effort, we propose to construct annotated dataset with two steps. Firstly, we collect entities from Baidu

[1] http://code.google.com/p/thebeast
[2] http://hazy.cs.wisc.edu/hazy/tuffy/
[3] http://www9-old.in.tum.de/people/jain/mlns/
[4] http://alchemy.cs.washington.edu/

Lexical Features
character(i,c+) $\wedge$ entityType(t+)$\Rightarrow$drop(i)
character(i,c+) $\wedge$ isNumber(i)$\Rightarrow$drop(i)
character(i,c+) $\wedge$ word(j,w+) $\wedge$ cwMap(i,j) $\Rightarrow$drop(i)
character(i,c+) $\wedge$ word(j,w+) $\wedge$ cwMap(i,j) $\wedge$ lastWord(j) $\Rightarrow$drop(i)
character(i,c+) $\wedge$ word(j,w+) $\wedge$ cwMap(i,j) $\wedge$ idf(j,v+) $\Rightarrow$drop(i)
character(i,c+) $\wedge$ word(j,w+) $\wedge$ cwMap(i,j) $\wedge$ entity(e+)$\Rightarrow$drop(i)
character(i,c+) $\wedge$ word(j,w+) $\wedge$ cwMap(i,j) $\wedge$ isCity(j) $\wedge$ entityType(e+) $\Rightarrow$drop(i)
character(i,c+) $\wedge$ word(j,w+) $\wedge$ cwMap(i,j) $\wedge$ isCity(j)$\wedge$ word(j,w1+) $\wedge$word(j+1,w2+) $\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j-1,w+) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+2,w+) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+3,w+) $\Rightarrow$ drop(i)
Distance and Position Features
character(i,c) $\wedge$ lenWord(wn+) $\wedge$ cwPosition(i,wp+) $\Rightarrow$drop(i)
character(i,c) $\wedge$ lenChar(cn+) $\wedge$ cwPosition(i,wp+) $\Rightarrow$drop(i)
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j,w+) $\wedge$ cwPosition(i,wp+)$\Rightarrow$drop(i)
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j,w+) $\wedge$ lenWord(wn+)$\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+) $\wedge$ cwPosition(i,wp+)$\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+) $\wedge$ cwPosition(i,wp+) $\wedge$ entityType(t+)$\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+2,w+) $\wedge$ cwPosition(i,wp+)$\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+2,w+) $\wedge$ cwPosition(i,wp+) $\wedge$ entityType(t+)$\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+3,w+) $\wedge$ cwPosition(i,wp+)$\Rightarrow$drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+3,w+) $\wedge$ cwPosition(i,wp+) $\wedge$ entityType(t+)$\Rightarrow$drop(i)
Features for Entity with Special Suffixes
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j,w+) $\wedge$ lenWord(l+) $\wedge$ entityType(t+) $\Rightarrow$ drop(i)
character(i,c+) $\wedge$ isCity(j) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+) $\wedge$ entityType(t+) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ isCity(j) $\wedge$ cwMap(i,j) $\wedge$ word(j,w1+) $\wedge$ word(j+1,w2+) $\wedge$ entityType(t+) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwPosition(i,p+) $\wedge$ $\neg$ isCity(j) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+) $\wedge$ (sufSchool(j+1) $\vee$ sufOrg(j+1) $\vee$ sufGov(j+1)) $\Rightarrow$ drop(i)
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+) $\wedge$ (sufSchool(j+1) $\vee$ sufOrg(j+1) $\vee$ sufGov(j+1)) $\Rightarrow$ drop(i)
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j-1,w+) $\wedge$ (sufSchool(j) $\vee$ sufOrg(j) $\vee$ sufGov(j)) $\Rightarrow$ drop(i)
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j-2,w+) $\wedge$ (sufSchool(j) $\vee$ sufOrg(j) $\vee$ sufGov(j)) $\Rightarrow$ drop(i)
character(i,c+) $\wedge$ cwMap(i,j) $\wedge$ word(j,w1+) $\wedge$ cwMap(ip,j-1) $\wedge$ city(ip,p) $\wedge$ (sufSchool(j+1) $\vee$ sufOrg(j+1) $\vee$ sufGov(j+1)) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j,w+) $\wedge$ entityType(t+) $\wedge$ $\neg$ isCity(j) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+1,w+)$\wedge$ entityType(t+) $\wedge$ $\neg$ isCity(j) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+2,w+)$\wedge$ entityType(t+) $\wedge$ $\neg$ isCity(j) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+3,w+)$\wedge$ entityType(t+) $\wedge$ $\neg$ isCity(j)) $\Rightarrow$ drop(i)
character(i,c) $\wedge$ cwMap(i,j) $\wedge$ word(j+4,w+)$\wedge$ entityType(t+) $\wedge$ $\neg$ isCity(j) $\Rightarrow$ drop(i)
cwMap(i,j) $\wedge$ word(j-1,w+) $\wedge$ isCity(j-1) $\wedge$ entityType(t+) $\Rightarrow$ drop(i)
cwMap(i,j) $\wedge$ (j=0) $\wedge$ word(j,w) $\wedge$ entityType(t+) $\Rightarrow$ drop(i)

Table 3: Descriptions of local formulas.

简称：A (abbreviation name: A)
A是E(的)?(中文)?简称 (A is the (Chinese)? abbreviation name of E)
E(的)?(中文)?简称(是)?A (the (Chinese)? abbreviation name of E is A)
A是E的同义词 (A is a synonym of E)
E和A是同义词 (E and A are synonyms)
A和E是同义词 (A and E are synonyms)
E represents the entity A represents the candidate abbreviation

Table 4: The lexical level regular expressions used to match entity and abbreviation pairs.

Baike[5], which is one of the most popular wiki-based Chinese encyclopedia and contains more than 6 millions items. Secondly, we use several simple regular expressions to extract abbreviation of entities from the crawled encyclopedia and snippets of search engine.

We crawled 3.2 millions articles from Baidu Baike. After that, we cleaned the HTML tags and extracted title, category and textual content from each articles. the structure of Baidu Baike is similar to that of Wikipedia, where titles are the name of the subject of the article, or may be a description of the topic. Hence, titles can be considered as the standard forms of entities. We select titles whose categories belong to location, organization, and facility to construct the standard forms list. It contains 302,633 items in total.

The next step is to use titles and corresponding articles to extract abbreviations. Through analyzing the dataset, we observe that most of abbreviations with the explicit description can be matched through a few of lexical level regular expressions. Table 4 shows the regular expressions we used in this work. Through this step, 30,701 abbreviation and entity pairs are extracted. We randomly select 500 pairs from them and manually check their correctness. The accuracy of the extracted pairs is around 98.2%.

To further increase the number of extractions, we propose to use Web as corpus and extract abbreviation and entity pairs from snippets of search engine results. For each entity whose abbreviation cannot be identified thorough the regular expressions described above, we combine entity and "简称 (abbreviation)" as queries for retrieving. The first three regular expressions in Table 4 are used to match abbreviation and entity pairs. Through this step, we get another 19,531 abbreviations. We also randomly select 500 pairs from them and manually check their correctness. The accuracy is around 95.2%. Finally, we merge the pairs extracted from Baike and search engine snippets and construct a list containing 50,232 abbreviation entity pairs. The accuracy of the list is 97.03%.

4.2 Experimental Settings

For evaluating the performance of the proposed method, we conducted experiments on the automatical constructed data. Total instances are randomly split with 75% for training, 5% for development and the other 20% for testing.

We compare the proposed method against start-of-the-art systems. Yang et al. (2009) proposed to use CRFs to model this. In this work, firstly, we re-implement the features they proposed. To fairly compare the two models, we also extend their work by including all local formulas we used in this work as features.

In our setting, we use FudanNLP [6] toolkit and thebeast [7] Markov Logic engine. FudanNLP is developed for Chinese natural language processing. We use the Chinese word segmentation of it under the default settings. The detailed setting of thebeast engine is as follows: The inference algorithm is the MAP inference with a cutting plane approach. For parameter learning, the weights for formulas is updated by an online learning algorithm with MIRA update rule. All the initial weights are set to zeros. The number of iterations is set to 10 epochs.

For evaluation metrics, we use precision, recall, and F-score to evaluate the performance of character deletion operation. To evaluate the performance of the entire generated abbreviations, we also propose to use accuracy to do it. It means that the generated abbreviation is considered as correct if all characters of its standard form are correctly classified.

[5]http://baike.baidu.com

[6]http://code.google.com/p/fudannlp

[7]http://code.google.com/p/thebeast

Methods	P	R	F	A
MLN-LF	83.2%	81.1%	82.1%	42.2%
MLN-LF+DPF	80.9%	84.3%	82.6%	45.7%
MLN-Local	82.4%	85.4%	83.9%	54.7%
MLN-Local+Global	81.6%	85.9%	83.7%	56.8%
CRFs-Yang	82.9%	83.6%	83.2%	39.7%
CRFs-LF	84.9%	83.7%	84.3%	40.5%
CRFs-LF+DPF	85.5%	83.5%	84.5%	40.6%
CRFs-Local	84.9%	83.8%	84.3%	40.8%

Table 5: The lexical level regular expressions used to match entity and abbreviation pairss.

4.3 Results

To evaluate the performance of our method, we set up several variants of the proposed method to compare with performances of CRFs. The *MLN-LF* method uses only the lexical features described in the Table 3. The *MLN-LF+DPF* method uses both lexical features and distance and position features. The *MLN-Local* method uses all local formulas described in the Table 3. The *MLN-Local+Global* methods combine both local formulas and global formulas together. For Yang's system, we use *CRFs-Yang* to represent the re-implemented method with feature set proposed by them and *CRFs-LF*, *CRFs-LF+DPF*, and *CRFs-Local* to represent feature sets similar as used by MLN.

Table 5 shows the performances of different methods. We can see that *MLN-Local+Global* achieve the best accuracy of entire abbreviation among all the methods. Although, the F-score of *MLN-Local+Global* is slightly worse than *MLN-Local*. We think that the global formulas contribute a lot for the entire accuracy. However, since the constraint of simultaneously dropping or keeping characters does not consider context, it may also bring some false matches. We can also see that, the methods modeled by MLN significantly outperform the performances of CRFs no matter which feature sets are used(base on a paired 2-tailed t-test with $p < 0.05$). We think that overfitting may be one of the main reasons.

From the perspective of entire accuracy, comparing the performances of *MLN-LF+DPF* and *MLN-Local*, we can see that features for entities with special suffixes contribute a lot. The relative improvement of *MLN-Local* is around 19.7%. It shows that the explicit rules are useful for improving the performance. However, these explicit rules only bring a small improvement to the accuracy of CRFs.

Comparing the performances of CRFs and MLNs, we can observe that CRFs achieve slightly better performance in classifying single characters. However MLNs achieve significantly better results of the entire accuracies. We think that these kinds of long distance features can be well handled by MLNs. These features are useful to capture the global constraints. Hence, MLNs can achieve better accuracy of the entire abbreviations.

In this paper, we also investigate the performance of different methods as the training data size are varied. Figure 1 shows the results. All full lines show the results of MLNs with different feature sets. The dot dash lines show the results of CRFs. From the results, we can observe that MLNs perform better than CRFs in most of cases. Except that MLNs with only lexical features work slightly worse than CRFs with small number of training data. From the figure, we also observe that the performance improvement of CRFs are not significant when the number of training data is larger than 35,000. However, methods using MLNs benefit a lot from the increasing data size. If more training instances are given, the performance of MLNs can still be improved.

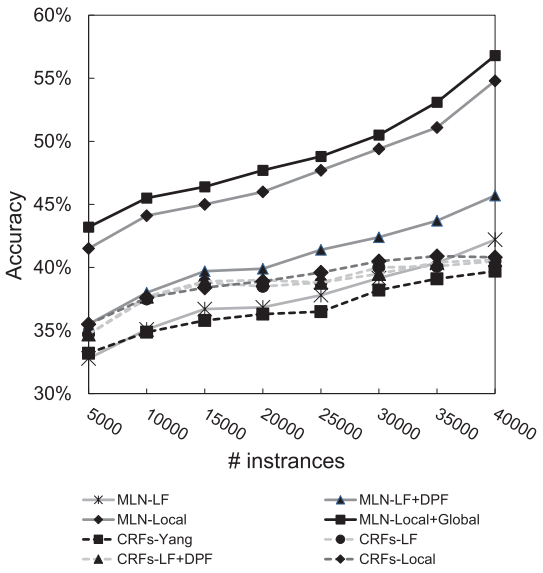

Figure 1: The impacts of training data size.

From the training procedures, we also empirically find that the training iterations of MLNs are small. It means that the convergence rate

of MLNs is fast. To evaluate the convergence rate, we also evaluate the dependence of the performances of MLNs on the number of training epochs. Figure 2 shows the results of *MLN-Local* and *MLN-Local+Global*. From the results, we can observe that the best performances can be achieved when the number of training epochs is more than nine. Hence, in this work, we set the number of iterations to be 10.

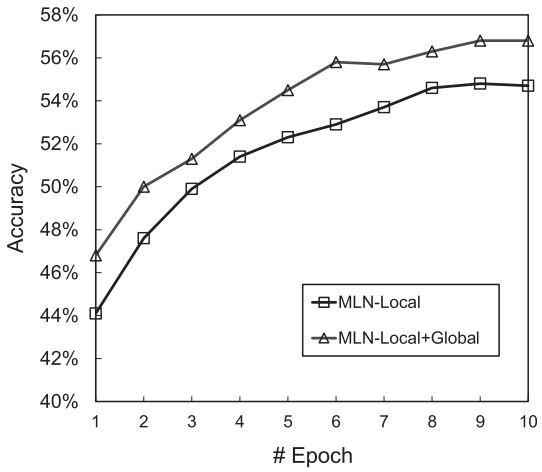

Figure 2: The performance curves on the number of training epochs.

5 Conclusions

In this paper, we focus on named entity abbreviation generation problem. We propose to use first-order logic to model rich linguistic features and global constraints. We convert the abbreviation generation to character deletion and keep operations. Linguistic features and relations between different operations are represented by local and global logic formulas respectively. Markov Logic Network frameworks is adopted for learning and predication. To reduce the human effort in constructing the training data, we also introduce an automatical training data construction methods with sample strategies. We collect standard forms of entities from online encyclopedia, use a few simple patterns to extract abbreviations from documents and search engine snippets with high precision as training data. Experimental results show that the proposed methods achieve better performance than state-of-the-art methods and can efficiently process large volumes of data.

6 Acknowledgement

The authors wish to thank the anonymous reviewers for their helpful comments. This work was partially funded by National Natural Science Foundation of China (61003092, 61073069), National Major Science and Technology Special Project of China (2014ZX03006005), Shanghai Municipal Science and Technology Commission (No.12511504502) and "Chen Guang" project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation(11CG05).

References

David Andrzejewski, Xiaojin Zhu, Mark Craven, and Benjamin Recht. 2011. A framework for incorporating general domain knowledge into latent dirichlet allocation using first-order logic. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1171–1177. AAAI Press.

AiTi Aw, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for sms text normalization. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, pages 33–40, Sydney, Australia, July. Association for Computational Linguistics.

Johan Bollen, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8.

Jing-shin Chang and Yu-Tso Lai. 2004. A preliminary study on probabilistic models for chinese abbreviations. In *Proceedings of the Third SIGHAN Workshop on Chinese Language Learning*, pages 9–16.

Jing-Shin Chang and Wei-Lun Teng. 2006. Mining atomic chinese abbreviations with a probabilistic single character recovery model. *Language Resources and Evaluation*, 40(3-4):367–374.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 368–378, Portland, Oregon, USA, June. Association for Computational Linguistics.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages

421–432, Stroudsburg, PA, USA. Association for Computational Linguistics.

Minlie Huang, Xing Shi, Feng Jin, and Xiaoyan Zhu. 2012. Using first-order logic to compress sentences. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Long Jiang, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 151–160, Portland, Oregon, USA, June. Association for Computational Linguistics.

Shangpu Jiang, D. Lowd, and Dejing Dou. 2012. Learning to refine an automatically extracted knowledge base using markov logic. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*, pages 912–917.

Zhifei Li and David Yarowsky. 2008a. Mining and modeling relations between formal and informal Chinese phrases from web corpora. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 1031–1040, Honolulu, Hawaii, October. Association for Computational Linguistics.

Zhifei Li and David Yarowsky. 2008b. Unsupervised translation induction for chinese abbreviations using monolingual corpora. In *Proceedings of ACL-08: HLT*, pages 425–433, Columbus, Ohio, June. Association for Computational Linguistics.

Cindy Xide Lin, Bo Zhao, Qiaozhu Mei, and Jiawei Han. 2010. Pet: a statistical model for popular events tracking in social communities. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 929–938, New York, NY, USA. ACM.

Fei Liu, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 1035–1044, Stroudsburg, PA, USA. Association for Computational Linguistics.

Xinfan Meng, Furu Wei, Xiaohua Liu, Ming Zhou, Sujian Li, and Houfeng Wang. 2012. Entity-centric topic-oriented opinion summarization in twitter. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 379–387, New York, NY, USA. ACM.

Naoaki Okazaki, Mitsuru Ishizuka, and Jun'ichi Tsujii. 2008. A discriminative approach to japanese abbreviation extraction. In *Proceedings of the Third International Joint Conference on Natural Language Processing (IJCNLP 2008)*, pages 889–894.

Hoifung Poon and Pedro Domingos. 2008. Joint unsupervised coreference resolution with markov logic. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 650–659, Stroudsburg, PA, USA. Association for Computational Linguistics.

Matthew Richardson and Pedro Domingos. 2006. Markov logic networks. *Machine Learning*, 62(1-2):107–136.

Sebastian Riedel and Ivan Meza-Ruiz. 2008. Collective semantic role labelling with markov logic. In *Proceedings of the Twelfth Conference on Computational Natural Language Learning*, CoNLL '08, pages 193–197, Stroudsburg, PA, USA. Association for Computational Linguistics.

Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web*, WWW '10, pages 851–860, New York, NY, USA. ACM.

P. Singla and P. Domingos. 2006. Entity resolution with markov logic. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 572–582.

Li-Xing Xie, Ya-Bin Zheng, Zhi-Yuan Liu, Mao-Song Sun, and Can-Hui Wang. 2011. Extracting chinese abbreviation-definition pairs from anchor texts. In *Machine Learning and Cybernetics (ICMLC)*, volume 4, pages 1485–1491.

Dong Yang, Yi-cheng Pan, and Sadaoki Furui. 2009. Automatic chinese abbreviation generation using conditional random field. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, NAACL-Short '09, pages 273–276, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dong Yang, Yi-Cheng Pan, and Sadaoki Furui. 2012. Vocabulary expansion through automatic abbreviation generation for chinese voice search. *Computer Speech & Language*, 26(5):321 – 335.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1*, ACL '09, pages 405–413, Stroudsburg, PA, USA. Association for Computational Linguistics.