Bootstrapping Large-scale Named Entities using URL-Text Hybrid Patterns

Chao Zhang **Shiqi Zhao** **Haifeng Wang**
Baidu Inc
No. 10, Shangdi 10th Street, Haidian District, Beijing 100085, China
{zhangchao01,zhaoshiqi,wanghaifeng}@baidu.com

Abstract

Automatically mining named entities (NE) is an important but challenging task, pattern-based and bootstrapping strategy is the most widely accepted solution. In this paper, we propose a novel method for NE mining using web document titles. In addition to the traditional text patterns, we propose to use url-text hybrid patterns that introduce url criterion to better pinpoint high-quality NEs. We also design a multiclass collaborative learning mechanism in bootstrapping, in which different patterns and different classes work together to determine better patterns and NE instances. Experimental results show that the precision of NEs mined with the proposed method is 0.96 and 0.94 on Chinese and English corpora, respectively. Comparison result also shows that the proposed method significantly outperforms a representative method that mines NEs from large-scale query logs.

1 Introduction

The task of named entity mining (NEM) aims to mine named entities (NE) of given categories from raw data. NEM is essential in many applications. For example, NEM can generate NE gazetteers necessary for the task of named entity recognition (NER) (Cohen and Sarawagi, 2004; Kazama and Torisawa, 2008; Talukdar et al., 2006). It can also help improve the search results in web search (Paşca, 2004), and increase the coverage of knowledge graphs.

Extensive research has been conducted on NEM, in which pattern-based methods are the most popular. Handcrafted or automatically learnt patterns are usually used to extract NE instances from various corpora, such as web documents, search engine's retrieved snippets, and query logs. Bootstrapping strategy is often applied to generate more patterns and instances iteratively so as to improve the coverage of the system.

The method we propose also belongs to the family of pattern-based NEM. However, our method is a departure from the previous ones. It makes contributions from the following aspects: First, we design url-text hybrid patterns instead of the traditional text patterns. We take url criterion into account, so as to measure the quality of the source webpages. Second, we propose Multiclass Collaborative Learning (MCL) mechanism, which globally scores and ranks the patterns and NE instances within mutiple classes in bootstrapping.

We evaluate our method in two languages, i.e., Chinese and English, so as to demonstrate the language-independent nature of the method. We mine NEs using the system for five categories in both languages, including *star*, *film*, *TV play*, *song*, and *PC game*. Experimental results show that the average precision of the extracted NEs is 96% in Chinese and 94% in English. Meanwhile, the average coverage computed against a benchmark repository is 61% and 55% for the two languages. Comparative experiments further show that our method significantly outperforms a representative conventional method.

2 Related Work

In this section, we review the previous studies on NEM from three aspects: the data resource used, the proposed methods, and particularly the bootstrapping strategy.

Various resources have been exploited for NEM. Many researchers make use of large-scale web corpora and learn NEs surrounded by certain context patterns (Paşca, 2004; Downey et al., 2007). Others mine NEs using web search engines. They submit extraction patterns as queries to search engines, and extract NEs matching the

International Joint Conference on Natural Language Processing, pages 293–301,
Nagoya, Japan, 14-18 October 2013.

patterns from the retrieved snippets (Etzioni et al., 2005; Etzioni et al., 2004; **?**; Kozareva and Hovy, 2010). There are also studies extracting NEs structured HTML tables (Dalvi et al., 2012). Besides web documents, NEs as well as their attributes can also be mined from search engine query logs, since many users tend to search for named entities in their queries (Paşca, 2007a; Paşca, 2007b).

As an alternative, this paper proposes to mine NEs from vertical websites titles, based on our observation that NEs of a class c can generally be found in webpage titles of some vertical websites of class c. Our statistics show that 99% out of $10,000$ random NEs appear in webpage titles. Besides, webpage titles have the advantage that they are of better quality than free-text documents, while less noisy than user queries.

Pattern-based methods are the most popular ones in NEM (Riloff and Jones, 1999; Thelen and Riloff, 2002; Etzioni et al., 2004; Paşca, 2004; Talukdar et al., 2006; Paşca, 2007b; Wang and Cohen, 2009; Kozareva and Hovy, 2010). NE extraction patterns in previous papers can be roughly classified into two types, i.e., Hearst patterns and class-specific wrappers. Hearst patterns are named after Hearst (1992), who among the first to design patterns, such as "E is a C", "C including E", to extract hyponyms / hypernyms. The surface patterns were later extended to lexico-syntactic patterns (Thelen and Riloff, 2002; Paşca, 2004), so that the pattern-filling instances can be identified more accurately via considered constraints.

Hearst patterns are binary patterns containing two slots. In contrast, class-specific wrappers are unary patterns with a single slot (Paşca, 2007b; Wang and Cohen, 2008). For example, the pattern "the film * was directed by" is a wrapper for the *film* class, in which the place holder "*" can be replaced by any film name. Wrappers need to be learnt for each NE class of interest. Our method proposed in this paper is also a pattern-based one. However, we design a novel type of url-text hybrid pattern, which not only benefits from the conventional textual wrappers, but also takes advantage of url constraint.

Most methods mentioned above are weakly-supervised, in which a few patterns, heuristic rules or instances are fed to the system as seeds, and the system enriches patterns and NE instances iteratively. Bootstrapping is widely used in these methods (Riloff and Jones, 1999; Thelen and Riloff, 2002; Paşca, 2007b; Wang and Cohen, 2008). The bootstrapping algorithm can effectively reduce manual intervention in building the system. However, it is prone to noise brought in during iterations. We therefore design a Multiclass Collaborative Learning (MCL, detailed in Section 3.4) mechanism in this paper, which guarantees the quality of the generated new patterns and instances by introducing inter-class and intra-class scoring criteria.

3 Named Entity Mining

3.1 Overview of the Method

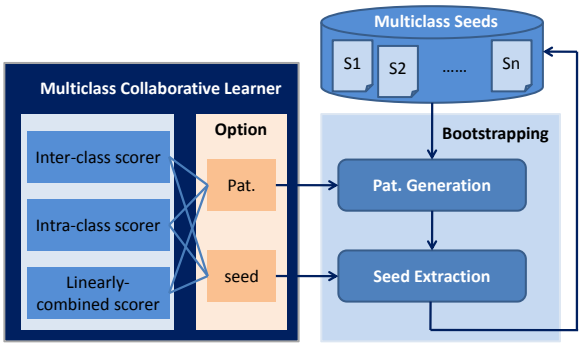

Figure 1: Framework of named entity mining

Named entities of a category are often organized in corresponding vertical websites, where each named entity is displayed in a single webpage. For example, it's easy to extract film NEs from IMDB[1] web titles with regular expressions.

Our method learns NE extraction patterns from web titles (text pattern) and introduces url constraint (url pattern) to make the extraction results more precise. As described in Figure 1, our method uses bootstrapping strategy in pattern generation and seed extraction. We also propose Multiclass Collaborative Learning (MCL) mechanism to filter noise introduced in iterations.

3.2 URL-Text Hybrid Patterns

3.2.1 Motivation

Text patterns are widely used as wrappers in tasks like information extraction and relation extraction. To improve the accuracy of wrappers, a lot of constraints such as part-of-speech tags (Etzioni et al., 2005) and trigger words (Talukdar et al., 2006) were introduced to tackle the

[1] www.imdb.com

tricky conditions. However, simple wrappers can also acquire high-quality NEs in specific conditions. For example, "^(.+?)$" is qualified to extract person name from the titles whose corresponding urls match the regular expression "http://www.nndb.com/people/\d+/\d+/".

Based on the consideration above, we take the quality of urls into consideration when using wrappers. We use simple text patterns if the websites are of high-quality, and have to use complicated text patterns if the website's quality is low. We therefore design url-text hybrid patterns to guarantee the capability of the patterns from both url and text aspects.

3.2.2 URL Patterns

Similar urls share the same pattern in many websites (Blanco et al., 2011). For example, all IMDB webpages describing video information match pattern "http://www.imdb.com/title/tt\d+/$". Therefore, we can take the url pattern as the identity of the website. Url patterns are globally learned using a large-scale url database. The process is as follows:

1. Given a url, we generate candidate url patterns by replacing the segments separated with "/" from its non-domain parts with slots respectively. For example, for url: "www.AAA.com/BBB/123" in which "/BBB/123" is the non-domain part, we can generate two candidate patterns "www.AAA.com/**SLOT**/123" and "www.AAA.com/BBB/**SLOT**".

2. All candidate patterns are accumulated on the url database. The ones with a frequency above a pre-defined threshold k are retained.

3. For each retained candidate pattern, we generate the final url pattern by replacing the slot with a regular expression based on the statistics of its slot fillers. For instance, for the candidate pattern "www.AAA.com/BBB/**SLOT**", if the slot can be filled with "123", "234", and "456", then the slot can be replaced with "\d+", meaning that this slot can be filled with any number sequence. Accordingly, the final url pattern should be "www.AAA.com/BBB/\d+".

3.2.3 Text Patterns

Text patterns are commonly used in NEM. Here we use a classical method to generate text patterns. Given a seed NE s in a category c, and a title t containing s, the text patterns are generated as:

1. Segment the title t into a word sequence.

2. Match the seed s in t, and replace s with the slot "(.+?)"[2].

3. Generate patterns that contain the slot as well as words preceding and succeeding the slot within a pre-defined window size. Several patterns can be yielded in this way given different window sizes. We set the window size to 2, 3, 4, and 5 in our experiment.

3.2.4 Hybrid Patterns

A url-text hybrid pattern (utp), combining both url and text patterns, is defined as a 4-tuple: $utp = (up, tp, c, f)$, where up and tp are the url pattern and text pattern respectively, c indicates the category that utp belongs to, and f (scored by Eq.(6)) denotes the confidence of utp.

We use UTP to denote a set of utps, and use UTP_i to denote the UTP of the i-th category c_i. A hybrid pattern is more strict than a url pattern and a text pattern separately. As we will show, the NEs it can extract are of better quality and coverage.

3.3 Bootstrapping

As described in Algorithm 1, our method $GenerateUTP$ generates raw patterns (r_UTP^k) with seeds ($Seeds^{k-1}$) from web titles (WT) in the k-th iteration. Likewise, raw NE instances (r_ins^k) are extracted by $ExtractNE$ in the following steps. $SelectUTP$ and $SelectNE$ output high quality patterns and NE instances respectively. These two functions are based on MCL mechanism described in Section 3.4. During these processes, each pattern is scored by Eq.(5) and is kept if its score is above a threshold, and the instances yielded in each iteration are ranked according to Eq.(6), and those ranked in the top $1/k$ are selected and added (with function $AddSeeds$) into the seed set.

We use $\#(ins^k)$ to denote the number of instances after the k-th iteration. The iterations will terminate if $\#(ins^k)/\#(ins^{k-1}) < \eta$, where η is

[2] "(.+?)" is a regular expression used to extract arbitrary strings

Algorithm 1 Bootstrapping for NE Mining

Require:
$Seeds^0$ for n categories: $\{S_1^0, S_2^0, \ldots, S_n^0\}$
webpage titles (WT);
iteration count $k = 1$;

Ensure:
1: **while** Terminate criterion is not met **do**
2: $r\_UTP^k = GenerateUTP(Seeds^{k-1}, WT)$;
3: $UTP^k = SelectUTP(r\_UTP^k, Seeds^{k-1}, WT)$;
4: $r\_ins^k = ExtractNE(UTP^k, WT)$;
5: $ins^k = SelectNE(r\_ins^k)$;
6: $Seeds^k = AddSeeds(ins^k, Seeds^{k-1})$;
7: $k = k + 1$;
8: **end while**
9: **return** ins^k:Named entities for n categories $\{NE_1, NE_2, \ldots, NE_n\}$;

a threshold ($\eta = 1.01$ in our experiments). All the extracted NEs after the last iteration are output along with their confidence score computed according to Equ.(6). One can set threshold w.r.t. the confidence score, so as to select high-quality named entities for certain applications.

3.4 Multiclass Collaborative Learning (MCL)

In this section, we design collaborative learning mechanism, which contains inter-class and intra-class scoring criteria, to better control the quality of the patterns and NE instances bootstrapped in iterations.

3.4.1 Inter-class Scoring

If an NE of category c_i can also be extracted with patterns from other categories, it is likely that it is noise, or at least is an ambiguous NE that is unsuitable to be used as a seed of c_i. Likewise, if a pattern of class c_i can also be generated by seeds from other categories, this pattern is obviously not a high-quality pattern for category c_i. Thus, we can score the patterns and seeds of the target category with the help of the other classes, which is termed "inter-class scoring".

The inter-class score for patterns is defined as:

$$P_1(c_i|utp) = \frac{P(c_i) \times P(utp|c_i)}{\sum_j P(c_j) \times P(utp|c_j)} \quad (1)$$

where: $P(c_i) = |S_i|/|\bigcup S_j|$, in which S_i denotes the seed set of category c_i and $|\cdot|$means the size of a set. During initialization, we prepare approximately the same number of seeds for each class. $P(utp|c_i) = |S_i(utp)|/|S_i|$, in which $S_i(utp)$ denotes the set of seeds in class c_i which generate the pattern utp.

The inter-class score for instances[3] is defined as:

$$P_1(c_i|s) = \frac{P(c_i) \times P(s|c_i)}{\sum_j P(c_j) \times P(s|c_j)} \quad (2)$$

where: $P(c_i)$ is defined as above, and $P(s|c_i) = \frac{Freq_i(s)}{\sum_{s' \in S_i} Freq_i(s')}$, $Freq_i(s)$ means the number of c_i's patterns that can extract instance s, S_i means all instances of category c_i.

3.4.2 Intra-class Scoring

Besides inter-class scoring, we also design an intra-class scoring criterion. The basic hypothesis is that, if a pattern generates a lot of instances that cannot be recalled by other patterns in this class, the pattern is likely to be incorrect.

For each class c_i, and the set of m patterns in the current iteration $UTP_i = utp_i^1, utp_i^2, \ldots, utp_i^m$, we compute the intra-class score for utp (say, utp is the j-th pattern utp_i^j) as:

$$P_2(c_i|utp) = \frac{|S_i(utp) \cap S_i^H|}{|S_i(utp)|} \quad (3)$$

where $S_i(utp)$ means the set of instances extracted by utp in class c_i, S_i^H is a set of high-quality instances extracted with all patterns in class c_i. Here "high-quality" is guaranteed by discarding the instances with frequency lower than a threshold T.

Likewise, intra-class scoring can also be defined for instances: the instances matching more patterns in class c_i are more likely to be correct instances of this class. The intra-class score for a seed s is computed as:

$$P_2(c_i|s) = \frac{|UTP_i(s)|}{|UTP_i|} \quad (4)$$

where $UTP_i(s)$ denotes the set of patterns in class c_i that can extract instance s, while UTP_i denotes the set of all patterns in c_i.

3.4.3 Linearly-combined Scoring

The final score for patterns and instances linearly combines both inter- and intra-class scores as follows:

[3] we also use s to denote an instance generated during bootstrapping.

For patterns:

$$P(c_i|utp) = \lambda P_1(c_i|utp) + (1-\lambda)P_2(c_i|utp) \quad (5)$$

For instances:

$$P(c_i|s) = \lambda P_1(c_i|s) + (1-\lambda)P_2(c_i|s) \quad (6)$$

In our experiments, λ is set to 0.5.

4 Evaluation

We evaluate our NE extraction method on five classes, i.e., *star*, *film*, *TV play*, *song*, and *PC game*. The reason to select these classes is that they are among the most frequently searched in search engines.

4.1 Experimental Data

In the experiments, we mainly evaluate the proposed method on Chinese. However, we also test the effectiveness of the method on English (Section 5.3). We therefore prepare experimental data for both languages.

We run our model on approximately 9.7 billion Chinese web titles and 13 billion English web titles respectively. Chinese web titles were collected from high-quality webpages after spam filtering and pageranking while English web titles were taken from all of our crawled English webpages. Note that, although the English corpus is larger than the Chinese one, it is still noisy and more sparse, given the fact that there are much more English (56.6%) webpages than Chinese (4.5%) on the whole internet[4]. The English titles are lowercased in preprocessing.

4.2 Metrics

We evaluate the methods based on precision (P), coverage (C), as well as the volume (V) of the extracted NEs. In particular, precision is defined as the percentage of correct NEs of a given class from the automatically extracted ones. Precision is manually evaluated, in which we randomly sample 100 NEs from each resulting NE set of a given class, and ask two annotators to independently annotate whether each extracted NE belongs to the target class. The samples with different annotations are then reviewed by both annotators to produce the final result.

Recall is more difficult to assess. Inspired by (Etzioni et al., 2004), we evaluate coverage against benchmark NE repositories. More specifically, we select a popular website for each given category in the corresponding language. For example, we use IMDB as the benchmark NE repository for categories *star*, *film* and *TV play* in English. All of the websites for constructing benchmark data on both Chinese (CH) and English (EN) are summarized in Table 1.

	Class	Website	Vol
CH	star	yule.sohu.com/star	4,643
	film	mtime.com	25,457
	TV play	www.mtime.com	4,080
	song	music.baidu.com	126,127
	PC game	pc.pcgames.com.cn	7,711
EN	star	imdb.com/	545,853
	film	imdb.com/	160,188
	tv play	imdb.com/	19,823
	song	spotify.com/	171,270
	pc game	gamespot.com/pc	8,131

Table 1: Benchmark dataset

The benchmark NEs were extracted from the websites using handcrafted patterns. Post-processing is done for the Chinese data, including discarding films and TV plays scored by only one viewer and songs played no more than 10 times. These filtering clues are extracted from the websites along with the NEs. For the English data, NEs are limited to those beginning with English characters and consisting of only English characters and some specific symbols ($'.-$ $:, !\#$). All English NEs are lowercased. The last column of Table 1 shows the statistics of the benchmark data. Coverage is computed as the percentage of NEs in a given benchmark set covered by the automatically extracted NEs. Please note that those websites for constructing benchmark data are not used in url patterns in the following experiments.

4.3 Results on Chinese

To extract Chinese NEs for the 5 examined classes, we first select 200 seeding NEs for each class. These seeds are randomly sampled from the top-5000 hot NEs for each class from Baidu[5] query logs. The results are shown in Table 2.

[4] http://en.wikipedia.org/wiki/Languages_used_on_the_Internet

[5] www.baidu.com, the largest Chinese search engine in the world.

Category	P	C	C_{hot}	Vol
star	0.99	0.85	0.90	7,630
film	0.95	0.76	0.86	24,183
TV play	0.92	0.82	0.93	21,655
song	0.96	0.12	0.33	11,011
PC game	0.96	0.50	0.75	14,049
average	0.96	0.61	0.75	15,706

Table 2: NEM results on the Chinese corpus

	star	film	TV play	song	PC game
CH	0.52	0.53	0.86	0.11	0.79
EN	0.78	0.82	0.87	0.78	0.84

Table 3: Percentage of NEs out of benchmark dataset

As can be seen from the Table 2, the precision of the extracted NEs is pretty high, which exceeds 0.92 on all five classes. On the other hand, the coverage varies across different classes. Especially, the coverage on songs is very low, which is only 0.12. After observing the extraction patterns, we found that the low coverage of songs is mainly due to the complexity of the patterns. Specifically, the titles of music websites usually contain not only the song's name, but also the singer's name. For example, the title "青花瓷-周杰伦-在线试听mp3下载-酷我音乐(*Green Flower Porcelain - Jay Chou - online audition mp3 download - kuwo music*)" is from a music website "www.kuwo.cn", in which "青花瓷(*Green Flower Porcelain*)" is the song's name, while "周杰伦(*Jay Chou*)" is the singer's name. The singer's name may seriously influence the generality of the induced text patterns.

We further evaluate our method's coverage of hot NEs. Here an NE is deemed hot if its daily search frequency is no less than 10 according to our query logs. The fourth column (C_{hot}) of Table 2 depicts the results. We can see that the coverage of hot NEs is evidently higher than that of random NEs for all five categories. The volume of extracted NEs for each class is listed in the last column of Table 2. Furthermore, row 1 of Table 3 depicts the percentage of the extracted Chinese NEs that are out of the benchmark dataset, from which we can see that our method actually mines a lot of NEs that are not covered by the benchmark data. This demonstrates the importance of extracting NEs from multiple websites.

4.4 Comparison Results

In this section, we compare our method with the method proposed by (Paşca, 2007b). Paşca's method is guided by a small set of seed instances for each class. The method extracts NEs from user queries in 5 steps: (1) generating query patterns matching the seed instances, (2) identifying candidate NEs using the patterns, (3) representing each candidate NE with a vector of patterns extracting it, (4) representing each class with a vector of patterns extracting its seeds, (5) computing the similarity between the representing vectors of each candidate NE and the class, and ranking the candidate NEs according to the similarity. Extracting NEs from query logs is a promising direction since search queries reflect the netizen's true requirements. In our experiments, we implement Paşca's method using our query log data, which contains a total of 100 million Chinese queries from Baidu search engine. The seeds used here are the same as in our method.

	Class	P@500	P@5k	P@all	C
Paşca's method	star	0.83	0.53	0.29	0.62
	film	0.95	0.79	0.73	0.10
	TV play	0.96	0.59	0.39	0.32
	song	0.98	0.92	0.86	0.12
	PC game	0.73	0.34	0.11	0.28
	average	0.89	0.63	0.48	0.29
Our method	star	0.98	0.98	0.99	0.85
	film	1.00	1.00	0.95	0.76
	TV play	0.99	1.00	0.92	0.82
	song	1.00	0.94	0.96	0.12
	PC game	1.00	0.97	0.96	0.50
	average	0.99	0.98	0.96	0.61

Table 4: Comparison with Paşca (2007b)'s method

We compare two methods based on precision at different numbers of extracted NEs, by annotating 100 NEs out of the first 500, $5k$ and all respectively, as well as coverage. The comparison results are shown in Table 4. We can find from the result that our method significantly outperforms Paşca's in both precision and coverage (**C**). Especially, the precision of NEs extracted by Paşca's method sharply decreases when lower ranked NEs are examined, whereas the quality of NEs extracted by our method seems quite stable.

5 Analysis

5.1 Analysis of Experiment Settings

This section analyzes the influence of the experimental settings. We first introduce the performance when using text pattern only, and then examine the contribution of the inter- and intra-class scoring in the MCL learning. Finally, we show how the performance varies with different number of iterations. Table 5 shows the P@500, P@$5k$ and P@all performance when only using text patterns. The precision seems relatively good but the coverage is generally low. The precision falls rapidly as the number of selected NEs grows except the category *film*. This table indicates that url patterns play an important role in our method, without which the quality of the extracted NEs cannot be guaranteed.

Class	P@500	P@5k	P@all	C
star	0.97	0.85	0.62	0.31
film	0.98	0.98	0.97	0.20
song	0.89	0.70	0.37	0.08
TV play	0.96	0.93	0.72	0.23
PC game	0.56	0.18	0.11	0.01

Table 5: Performance when using text pattern only

Table 6 shows the P@500 and P@$5k$ performance of our method when we only use intra-class or inter-class scoring in MCL learning. We can find that there is a dramatic decrease in the performance in both settings, suggesting that both inter-class and intra-class scoring criterion are necessary to guarantee the accuracy of the extracted NEs, and they should be used together.

Table 7 shows the performance after 1, 3, and 5 iterations. The number of url patterns is also listed along with precision and coverage. As can be seen, the average precision only slightly drops from 0.97 to 0.96 after 5 iterations, whereas the average coverage increases significantly from 0.53 to 0.61. This is mainly because the extraction sources grow almost 3 times, from 314 url-patterns to 1129 for each category on average.

5.2 Error Analysis

We have analyzed the erroneous NEs extracted by our method. This paragraph analyzes errors regarding precision while the following paragraph describes errors about recall. It turns out that ambiguity is a main reason for the errors. We find

	Category	P@500	P@5k
Using only intra-class estimator in MCL	star	0.11	0.36
	film	0.17	0.18
	TV play	0.12	0.43
	song	0.09	0.12
	PC game	0.43	0.10
	average	0.18	0.24
Using only inter-class estimator in MCL	star	0.07	0.08
	film	0.97	0.97
	TV play	0.98	0.96
	song	0.25	0.35
	PC game	0.75	0.79
	average	0.60	0.63

Table 6: Performance when using only intra-class or inter-class scoring in MCL

it quite common that an NE belongs to more than one class. For example, a TV play might be adapted from a novel with the same name, a biographical film might be named after the protagonist, etc. Statistics reveal that in "www.mtime.com", which is the benchmark data for extracting Chinese films and TV plays in our work, 12.8% of the TV plays have homonymic films in the same website, while the percentage is 14% in its English counterpart "www.imdb.com". Our method suffers from the ambiguity problem since the homonymic NEs might yield url-text patterns belonging to other classes, and thereby bring in noisy NEs.

Besides, as pointed out in Section 4.3, title complexity is the main problem that hinders NE extraction. Particularly, some titles contain more than one NE, which makes it difficult to induce text-patterns for a certain class from these titles. Another reason leading to the mismatch of benchmark NEs is that some NEs have different forms in different sources. For instance, we extract the song "*New Years Project*", but the correct form in the benchmark data is "*New Year's Project*".

5.3 Language Adaptation

Our method is language-independent. This section presents the evaluation in English. The English data is described in Section 4.1. Table 8 shows the performance of our method.

We can see from the table that the precision of the extracted English NEs is also high. Compared with Table 2 above, we can find that the coverage of the English NEs is lower than that on Chinese. However, the volume of the extracted NEs is al-

#Iteration	measure	star	film	TV play	song	PC game	average
1	Precision	0.98	0.98	0.99	0.98	0.92	0.97
	Coverage	0.86	0.55	0.65	0.12	0.47	0.53
	url-text patterns	296	437	387	243	207	314
3	Precision	0.98	0.96	0.95	0.97	0.91	0.95
	Coverage	0.89	0.75	0.66	0.12	0.53	0.59
	url-text patterns	745	2,041	1,525	324	659	1,059
5	Precision	0.99	0.95	0.92	0.96	0.96	0.96
	Coverage	0.85	0.76	0.82	0.12	0.50	0.61
	url-text patterns	791	2,101	1,593	325	835	1,129

Table 7: Performance for varying number of iterations

Category	P	C	Vol
star	0.98	0.65	1,589,002
film	0.92	0.40	352,152
TV play	0.89	0.59	71,273
song	0.95	0.31	240,335
PC game	0.97	0.80	29,166
average	0.94	0.55	456,386

Table 8: Performance on English corpora

most 30 times larger. This is unsurprising, since there are much more NEs written in English than in Chinese on the internet. Given that the English corpus used in our experiments is only 1.4 times larger than the Chinese corpus, we believe that data sparseness might be a major cause of the low coverage. Likewise, from the row 2 of Table 3, our method also acquires a large proportion of NEs in English which do not exist in the benchmark websites.

To have a better understanding of the coverage problem, we examined the cases not extracted with our method. As we have analyzed the problem of song, we randomly sampled 1000 NEs missed by our method for the other 3 classes with low coverage on the Chinese test dataset, i.e., *star*, *film*, and *TV play*. We then examined whether the missed cases contain a lot of hot NEs according to the following heuristics: (1) If a star has no picture on the imdb page, then it should not be deemed hot. Our statistics show that 97.3% missed stars have no pictures. (2) If a film's duration is no longer than one hour and the number of viewers grading it on IMDB is less than 10, then the film should not be hot. 90.6% missed films are not hot according to this criterion. (3) Similar to film, if the number of reviewers grading a TV play is less than 10, then it is not hot. 69.9% of the missed TV plays are not hot accordingly. On the whole, the above numbers suggest that the NEs not covered by our method are mostly unpopular ones, which may seldom be used in real applications.

6 Conclusion and Future Work

In this paper, we propose to extract NEs from web document titles using url-text hybrid patterns. A multiclass collaborative learning mechanism is introduced into the bootstrapping algorithm to better perform the quality control. We evaluate our method on five categories popular in real applications, in both Chinese and English. The results reveal that the precision and coverage (against benchmark data) of the extracted NEs are 0.96 / 0.61 in Chinese, and 0.94 / 0.55 in English. Detailed analysis demonstrates that the url-text hybrid patterns are superior to conventional text wrappers, and the multiclass collaborative learning mechanism is effective. Further comparison also shows that our method can significantly outperform a representative method that learns NEs from query logs.

Our future work will be carried out along two directions, i.e. improving the text-pattern induction approach and testing the method in more other languages.

7 Acknowledgment

This work is supported by National High-tech R&D Program of China (863 Program) under the grant number: 2011AA01A207. We give warm thanks to Prof. Jian-Yun Nie and other anonymous reviewers for their comments.

References

Lorenzo Blanco, Nilesh Dalvi, and Ashwin Machanavajjhala. 2011. Highly efficient algorithms for structural clustering of large websites. In *Proceedings of the 20th international conference on World wide web*, WWW '11, pages 437–446, New York, NY, USA. ACM.

William W. Cohen and Sunita Sarawagi. 2004. Exploiting dictionaries in named entity extraction: combining semi-markov extraction processes and data integration methods. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '04, pages 89–98, New York, NY, USA. ACM.

Bhavana Bharat Dalvi, William W. Cohen, and Jamie Callan. 2012. Websets: extracting sets of entities from the web using unsupervised information extraction. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 243–252, New York, NY, USA. ACM.

Doug Downey, Matthew Broadhead, and Oren Etzioni. 2007. Locating complex named entities in web text. In *Proceedings of the 20th international joint conference on Artifical intelligence*, IJCAI'07, pages 2733–2739, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Oren Etzioni, Michael Cafarella, Doug Downey, Stanley Kok, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2004. Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 100–110, New York, NY, USA. ACM.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. *Artif. Intell.*, 165(1):91–134, June.

Jun'ichi Kazama and Kentaro Torisawa. 2008. Inducing gazetteers for named entity recognition by large-scale clustering of dependency relations. In *ACL*, pages 407–415.

Zornitsa Kozareva and Eduard Hovy. 2010. A semi-supervised method to learn and construct taxonomies using the web. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 1110–1118, Stroudsburg, PA, USA. Association for Computational Linguistics.

Marius Paşca. 2004. Acquisition of categorized named entities for web search. In *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, CIKM '04, pages 137–145, New York, NY, USA. ACM.

Marius Paşca. 2007a. Organizing and searching the world wide web of facts – step two: harnessing the wisdom of the crowds. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 101–110, New York, NY, USA. ACM.

Marius Paşca. 2007b. Weakly-supervised discovery of named entities using web search queries. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, CIKM '07, pages 683–690, New York, NY, USA. ACM.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference innovative applications of artificial intelligence*, AAAI '99/IAAI '99, pages 474–479, Menlo Park, CA, USA. American Association for Artificial Intelligence.

Partha Pratim Talukdar, Thorsten Brants, Mark Liberman, and Fernando Pereira. 2006. A context pattern induction method for named entity extraction. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X 06, pages 141–148, Stroudsburg, PA, USA. Association for Computational Linguistics.

Michael Thelen and Ellen Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing - Volume 10*, EMNLP '02, pages 214–221, Stroudsburg, PA, USA. Association for Computational Linguistics.

Richard C. Wang and William W. Cohen. 2008. Iterative set expansion of named entities using the web. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 1091–1096, Washington, DC, USA. IEEE Computer Society.

Richard C. Wang and William W. Cohen. 2009. Automatic set instance extraction using the web. 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 441–449, Stroudsburg, PA, USA. Association for Computational Linguistics.