A Probabilistic Co-Bootstrapping Method for Entity Set Expansion

Bei Shi, Zhengzhong Zhang
Institute of Software,
Chinese Academy of Sciences,
Beijing, China

Le Sun, Xianpei Han
State Key Laboratory of Computer Science,
Institute of Software,
Chinese Academy of Sciences,
Beijing, China

{shibei, zhenzhong, sunle, xianpei}@nfs.iscas.ac.cn

Abstract

Entity Set Expansion (ESE) aims at automatically acquiring instances of a specific target category. Unfortunately, traditional ESE methods usually have the expansion boundary problem and the semantic drift problem. To resolve the above two problems, this paper proposes a probabilistic Co-Bootstrapping method, which can accurately determine the expansion boundary using both the positive and the discriminant negative instances, and resolve the semantic drift problem by effectively maintaining and refining the expansion boundary during bootstrapping iterations. Experimental results show that our method can achieve a competitive performance.

1 Introduction

Entity Set Expansion (ESE) aims at automatically acquiring instances of a specific target category from text corpus or Web. For example, given the capital seeds {*Rome*, *Beijing*, *Paris*}, an ESE system should extract all other capitals from Web, such as *Ottawa*, *Moscow* and *London*. ESE system has been used in many applications, e.g., dictionary construction (Cohen and Sarawagi, 2004), word sense disambiguation (Pantel and Lin, 2002), query refinement (Hu et al., 2009), and query suggestion (Cao et al., 2008).

Due to the limited supervision provided by ESE (in most cases only 3-5 seeds are given), traditional ESE systems usually employ bootstrapping methods (Cucchiarelli and Velardi, 2001; Etzioni et al., 2005; Pasca, 2007; Riloff and Jones, 1999; Wang and Cohen, 2008). That is, the entity set is iteratively expanded through a pattern generation step and an instance extraction step. Figure 1(a) demonstrates a simple bootstrapping process.

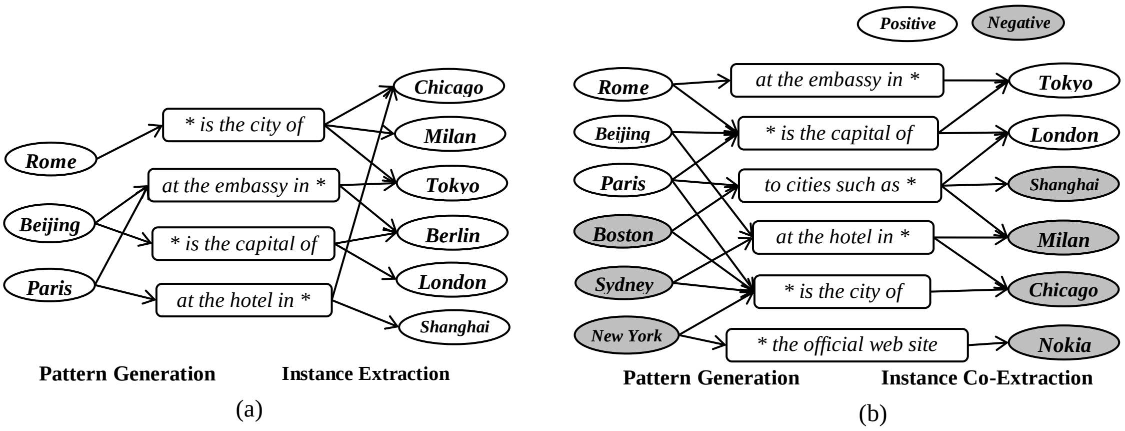

Figure 1: A demo of Bootstrapping (a) and Co-Bootstrapping (b)

Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers,
pages 2280–2290, Dublin, Ireland, August 23-29 2014.

However, the traditional bootstrapping methods have two main drawbacks:

1) **The expansion boundary problem**. That is, using only positive seeds (i.e., some example entities from the category we want to expand), it is difficult to represent which entities we want to expand and which we don't want. For example, starting from positive seeds {*Rome*, *Beijing*, *Paris*}, we can expand entities at many different levels, e.g., all capitals, all cities, or even all locations. And all these explanations are reasonable.

2) **The semantic drift problem**. That is, the expansion category may change gradually when noisy instances/patterns are introduced during the bootstrapping iterations. For example, in Figure 1 (a), the instance *Rome* will introduce a pattern "* *is the city of*", which will introduce many noisy city instances such as *Milan* and *Chicago* for the expansion of *Capital*. And these noisy cities in turn will introduce more city patterns and instances, and finally will lead to a semantic drift from *Capital* to *City*.

In recent years, some methods (Curran et al, 2007; Pennacchiotti and Pantel, 2011) have exploited mutual exclusion constraint to resolve the semantic drift problem. These methods expand multiple categories simultaneously, and will determine the expansion boundary based on the mutually exclusive property of the pre-given categories. For instance, the exclusive categories *Fruit* and *Company* will be jointly expanded and the expansion boundary of {*Apple, Banana, Cherry*} will be limited by the expansion boundary of {*Google, Microsoft, Apple Inc.*}. These methods, however, still have the following two drawbacks:

1) These methods require that the expanded categories should be mutually exclusive. However, in many cases the mutually exclusive assumption does not hold. For example, many categories hold a hyponymy relation (e.g., the categories *City* and *Capital*, because the patterns for *Capital* are also the patterns for *City*) or a high semantic overlap (e.g., the categories *Movies* and *Novels*, because some movies are directly based on the novels of the same title.).

2) These methods require the manually determination of the mutually exclusive categories. Unfortunately, it is often very hard for even the experts to determine the categories which can define the expansion boundaries for each other. For example, in order to expand the category *Chemical Element*, it is difficult to predict its semantic drift towards *Color* caused by the ambiguous instances {*Silver*, *Gold*}.

In this paper, to resolve the above problems, we propose a probabilistic Co-Bootstrapping method. The first advantage of our method is that we propose a method to better define the expansion boundary using both the positive and the discriminant negative seeds, which can both be automatically populated during the bootstrapping process. For instance, in Figure 1(b), in order to expand *Capital*, the Co-Bootstrapping algorithm will populate both positive instances from the positive seeds {*Rome, Beijing, Paris*}, and negative instances from the negative seeds {*Boston, Sydney, New York*}. In this way the expansion boundary of *Capital* can be accurately determined.

The second advantage of our method is that we can maintain and refine the expansion boundary during bootstrapping iterations, so that the semantic drift problem can be effectively resolved. Specifically, we propose an effective scoring algorithm to estimate the probability that an extracted instance belongs to the target category. Based on this scoring algorithm, this paper can effectively select positive instances and discriminant negative instances. Therefore the expansion boundary can be maintained and refined through the above jointly expansion process.

We have evaluated our method on the expansion of thirteen categories of entities. The experimental results show that our method can achieve 6%~15% P@200 performance improvement over the baseline methods.

This paper is organized as follows. Section 2 briefly reviews related work. Section 3 defines the problem and proposes a probabilistic Co-Bootstrapping approach. Experiments are presented in Section 4. Finally, we conclude this paper and discuss some future work in Section 5.

2 Related Work

In recent years, ESE has received considerable attentions from both research (An et al., 2003; Cafarella et al., 2005; Pantel and Ravichandran, 2004; Pantel et al., 2009; Pasca, 2007; Wang and Cohen, 2008) and industry communities (e.g., *Google Sets*). Till now, most ESE systems employ bootstrapping methods, such as *DIPRE* (Brin, 1998), *Snowball* (Agichtein and Gravano, 2000), etc.

The main drawbacks of the traditional bootstrapping methods are the expansion boundary problem and the semantic drift problem. Currently, two strategies have been exploited to resolve the semantic drift problem. The first is the ranking based approaches (Pantel and Pennacchiotti, 2006; Talukdar et al., 2008), which select highly confident patterns and instances through a ranking algorithm, with the assumption that high-ranked instances will be more likely to be the instances of the target category. The second is the mutual exclusion constraint based methods (Curran et al., 2007; McIntosh and Curran, 2008; Pennacchiotti and Pantel, 2011; Thelen and Riloff, 2002; Yangarber et al., 2002), which expand multiple categories simultaneously and determine the expansion boundary based on the mutually exclusive property of the pre-given categories.

3 The Co-Bootstrapping Method

3.1 The Framework of Probabilistic Co-Bootstrapping

Given the initial positive seeds and negative seeds, the goal of our method is to extract instances of a specific target semantic category. For demonstration, we will describe our method through the running example shown in Figure 1(b).

Specifically, Figure 2 shows the framework of our method. The central tasks of our Co-Bootstrapping method are as follows:

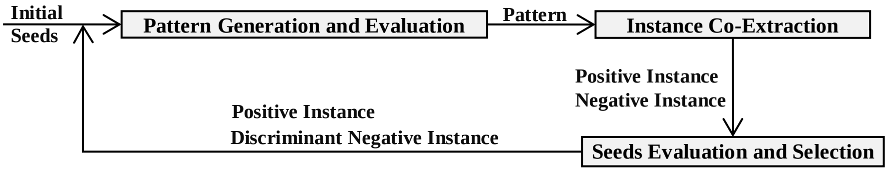
Initial Seeds
Pattern Generation and Evaluation
Pattern
Instance Co-Extraction
Positive Instance
Negative Instance
Positive Instance
Discriminant Negative Instance
Seeds Evaluation and Selection

Figure 2: The framework of probabilistic Co-Bootstrapping

1) **Pattern Generation and Evaluation.** This step generates and evaluates patterns using the statistics of the positive and the negative instances. Specifically, we propose three measures of pattern quality: the Generality (*GE*), the Precision of Extracted Instances (*PE*) and the Precision of Not Extracted Instances (*PNE*).

2) **Instance Co-Extraction.** This step co-extracts the positive and the negative instances using highly confident patterns. Specifically, we propose an effective scoring algorithm to estimate the probability that an extracted instance belongs to the target category based on the statistics and the quality of the patterns which extract it.

3) **Seed Selection.** This step selects the high ranked positive instances and discriminant negative instances to refine the expansion boundary by measuring how well a new instance can be used to define the expansion boundary.

The above three steps will iterate until the number of extracted entities reaches a predefined threshold. We describe these steps as follows.

3.2 Pattern Generation and Evaluation

In this section, we describe the pattern generation and evaluation step. In this paper, each pattern is a 4-grams lexical context of an entity. We use the Google Web 1T corpus's (Brants and Franz, 2006) 5-grams for both the pattern generation and the instance co-extraction in ESE. Our method generates patterns through two steps: 1) Generate candidate patterns by matching seeds with the 5-grams. 2) Evaluate the quality of the patterns.

For the first step, we simply match each seed instance with all 5-grams, then we replace the matching instance with wildcard "*" to generate the pattern.

Count	**Positive**	**Negative**
Extracted	Extracted Positive (***ep***)	Extracted Negative (***en***)
Not Extracted	Not Extracted and Positive (***nep***)	Not Extracted and Negative (***nen***)

(a)

Extracted Positive (***ep***)	*London*
Extracted Negative (***en***)	*Shanghai, Milan*
Not Extracted Positive (***nep***)	*Tokyo*
Not Extracted Negative (***nen***)	*Chicago, Nokia*

(b)

Table 1: (a) shows the four classes of instances according to polarity and extraction. (b) shows the four classes of the instances given *"to cities such as *"*

For the second step, we propose three measures to evaluate the quality of a pattern, correspondingly the *Generality (GE)*, the *Precision of Extracted Instances (PE),* and the *Precision of Not Extracted Instances (PNE)*. Specifically, given a pattern, we observed that all instances can be categorized into four classes, according to whether they belong to the target category and whether they can be extracted by the pattern (shown in Table 1(a)). For example, given the pattern *"to cities such as *"* in Figure 1(b), the instances under its four classes are shown in Table 1 (b).

The proposed three measures of the quality of a pattern can be computed as follows (In most cases, we cannot get the accurate number of *ep*, *en*, *nep* and *nen*. So this paper uses the corresponding known instances in the previous iteration to approximately compute *ep*, *en*, *nep* and *nen*):

1)**Generality** (*GE*). The Generality of a pattern measures how many entities can be extracted by it. A more general pattern will cover more entities than a more specific pattern. Specifically, the *GE* of a pattern is computed as:

$$GE = \frac{ep + en}{ep + en + nep + nen}$$

That is, the proportion of the instances which can be extracted by the pattern in the previous iteration.

2)**Precision of Extracted Instances** (*PE*). The *PE* measures how likely an instance extracted by a pattern will be positive. That is, a pattern with higher *PE* will be more likely to extract positive instances than a lower *PE* pattern. The *PE* is computed as:

$$PE = \frac{ep}{ep + en}$$

That is, the proportion of positive instances within all instances which can be extracted by the pattern in the previous iteration.

3)**Precision of Not Extracted Instances** (*PNE*). The *PNE* measures how likely a not extracted instance is positive. Instances not extracted by a high *PNE* pattern will be more likely to be positive. *PNE* is computed as:

$$PNE = \frac{nep/(ep + nep)}{nep/(ep + nep) + nen/(en + nen)}$$

Because the number of negative instances is usually much larger than the number of positive instances, we normalize the number of positive and negative instances in the formula.

Table 2 shows these measures of some selected patterns evaluated using the Google Web 1T corpus. We can see that the above measures can effectively evaluate the quality of patterns. For instance, *GE("* is the city of")=0.566* is larger than *GE*(*"*at the embassy in **"*)*=0.340,* which is consistent with our intuition that the pattern *"* is the city of"* is more general than *"at the embassy in *". PE*(*"* is the capital of"*)*=0.928* is larger than *PE*(*"* is the city of"*)*=0.269,* which is consistent with our intuition that the instances extracted by *"* is the capital of"* are more likely *Capital* than by*"* is the city of"*.

	GE	***PE***	***PNE***
*at the embassy in **	*0.340*	*0.833*	*0.312*
** is the capital of*	*0.321*	*0.928*	*0.224*
*to cities such as **	*0.426*	*0.875*	*0.566*
*at the hotel in **	*0.333*	*0.192*	*0.571*
** is the city of*	*0.566*	*0.269*	*0.592*
** the official web site*	*0.218*	*0.230*	*0.607*

Table 2: The *GE*, *PE* and *PNE* of some selected patterns

3.3 Instance Co-Extraction

In this section, we describe how to co-extract positive instances and discriminant negative instances. Given the generated patterns, the central task of this step is to measure the likelihood of an instance to be positive. The higher the likelihood, the more likely the instance belongs to the target category. To resolve the task, we propose a probabilistic method which predicts the probability of an instance to be positive, i.e., the *Instance Positive Probability* and we denote it as *P+*. Generally, the ***P+*** is determined by both the statistics and the quality of patterns. We start with the observation that:

1) If an instance is extracted by a pattern with a high *PE*, the instance will have a high ***P+***.

2) If an instance is not extracted by a high *PNE* pattern, the instance will have a high ***P+***.

3) If an instance is extracted by many patterns with high *PE* and not extracted by many patterns with high *PNE*, the instance will have a high ***P+***, and vice versa.

Based on the above observations, the computation of ***P+*** is as follows:

The Situation of One Pattern

For the situation that only one pattern exists, the ***P+*** of an instance can be simply computed as:

$$P{+}(e) = \begin{cases} PE(p) & \text{when } p \text{ extracts } e \\ PNE(p) & \text{otherwise} \end{cases}$$

where *e* denotes an extracted instance and *p* denotes a pattern which extracts *e*. This formula means that if the instance is extracted by a pattern, the ***P+*** is determined by the *PE* of the pattern. For example, in Figure 3 (a), the instance *Tokyo* is only extracted by the pattern *"at the embassy in *"* and the ***P+*** is determined by the *PE* of *"at the embassy in *"*, i.e., ***P+***(*Tokyo*)=*PE*(*"at the embassy in *"*).

The above formula also means when the instance cannot be extracted by the only pattern, the ***P+*** will be determined by the *PNE* of the pattern. For example, in Figure 3 (b), the instance *Tokyo* is not extracted by the only pattern *"at the hotel in *"* and the ***P+*** is only determined by the *PNE* of *"at the hotel in *"*, that is, ***P+****(Tokyo)=PNE*(*"at the hotel in *"*).

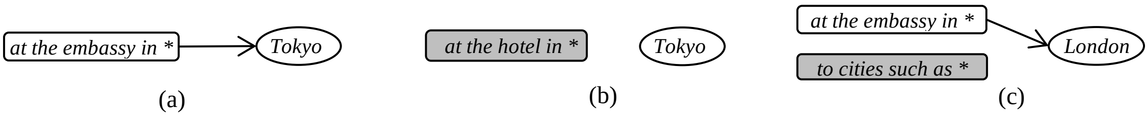

Figure 3: (a) *Tokyo* is extracted by *"at the embassy in *"*. (b) *Tokyo* is not extracted by *"at the hotel in *"*. (c) *London* is extracted by *"at the embassy in *"* and not extracted by *"to cities such as *"*.

The Situation of Multiple Patterns

In this section, we describe how to compute ***P+*** in the situation of multiple patterns. Specifically, we assume that an instance is extracted by different patterns independently. Therefore, given all the pattern-instance relations (i.e., whether a specific pattern extracts a specific instance), the likelihood for an instance *e* being positive is computed as:

$$PosLikelihood(e) \propto \prod_{p \in R^+} P(p \rightarrow e, e \in I^+) \prod_{p \in R^-} P(p \nrightarrow e, e \in I^+)$$

where R^+ is all the patterns which extract *e*, and R^- is all the patterns which do not extract *e*. I^+ is the set of all positive instances. $P(p \rightarrow e, e \in I^+)$ is the probability of the event *"pattern p extracts instance e and e is positive"*. Using Bayes rule, this probability can be computed as:

$$P(p \rightarrow e, e \in I^+) = P(p \rightarrow e)P(e \in I^+ | p \rightarrow e)$$

where $P(p \rightarrow e)$ is the probability of the event *"p extracts an instance e"*, its value is *GE(p)*; $P(e \in I^+ | p \rightarrow e)$ is the conditional probability that *e* is positive under the condition *"p extracts e"*, and its value is *PE(p)*. Finally $P(p \rightarrow e, e \in I^+)$ is computed as:

$$P(p \rightarrow e, e \in I^+) = GE(p)PE(p)$$

$P(p \nrightarrow e, e \in I^+)$ is the probability of the event "*p does not extract e and e is positive*", which can be computed as:

$$P(p \nrightarrow e, e \in I^+) = P(p \nrightarrow e)P(e \in I^+ | p \nrightarrow e)$$

$P(p \nrightarrow e)$ is the probability of *p* not extracting an instance *e*, and its value is 1-*GE(p)*. $P(e \in I^+ | p \nrightarrow e)$ is the conditional probability that *e* is positive under the condition *"p does not extract e"*, and its value is *PNE(p)*. Then $P(p \nrightarrow e, e \in I^+)$ is finally computed as:

$$P(p \nrightarrow e, e \in I^+) = (1 - GE(p))PNE(p)$$

For example, in Figure 3 (c), the instance *London* is extracted by the pattern *"at the embassy in *"* and not extracted by the pattern *"to cities such as *"*. In this situation, *PosLikelihood*(*London*)= [*GE*(*"at the embassy in *"*) × *PE*(*"at the embassy in"*)] × [(1-*GE*(*"to cities such as *"*)) × *PNE*(*"to cities such as *"*)].

Using the same intuition and the same method, the likelihood of an instance being negative is computed as:

$$NegLikelihood(e) \propto \prod_{p \in R^{+}} P(p \rightarrow e, e \notin I^{+}) \prod_{p \in R^{-}} P(p \nrightarrow e, e \notin I^{+})$$

where $P(p \rightarrow e, e \notin I^{+})$ is the probability of the event *"p extracts e and e is negative"*, which is computed as:

$$P(p \rightarrow e, e \notin I^{+}) = GE(p)(1 - PE(p))$$

$P(p \nrightarrow e, e \notin I^{+})$ is the probability of the event *"p does not extract e and e is negative"*, which is computed as:

$$P(p \nrightarrow e, e \notin I^{+}) = (1 - GE(p))(1 - PNE(p))$$

For instance, in Figure 3 (c), *NegLikelihood*(*London*) = [*GE*("*at the embassy in **") × (1-*PE*("*at the embassy in*"))] × [(1-*GE*("*to cities such as **")) × (1-*PNE*("*to cities such as **"))].

Finally, the *Instance Positive Probability*, ***P+***, is computed as:

$$P{+}(e) = \frac{PosLikelihood(e)}{PosLikelihood(e) + NegLikelihood(e)}$$

3.4 Seed Selection

In this section, we describe how to select positive and discriminant negative instances at each iteration.

To determine whether an instance is positive, we use a threshold of ***P+*** to determine the polarity of instances, which can be empirically estimated from data. The instances which have much higher ***P+*** than the threshold will be added to the set of positive instances. For example, *London* and *Tokyo* in Figure 1 (b) are selected as positive instances.

To select discriminant negative instances, we observed that not all negative instances are the same useful for the expansion boundary determination. Intuitively, the discriminant negative instances are those negative instances which are highly overlapped with the positive instances. For instance, due to the lower overlap between categories *Fruit* and *Capital*, *Apple* is not a discriminant negative instance since it provides little information for the expansion boundary determination. Therefore, the instances near the threshold are used as the discriminant negative instances in the next iteration. (Notice that, the computation of *GE*, *PE* and *PNE* still uses all positive and negative instances, rather than only discriminant negative instances). For example, in Figure 1(b), *Shanghai*, *Milan* and *Chicago* are selected as discriminate negative instances, and *Nokia* will be neglected. Finally the boundary between *Capital* and *City* can be determined by the positive instances and the discriminant negative instances.

4 Experiments

4.1 Experimental Settings

Category	Description	Category	Description
CAP	Place: capital name	**FAC**	Facilities: names of man-made structures
ELE	chemical element	**ORG**	Organization: e.g. companies, governmental
FEM	Person: female first name	**GPE**	Place: Geo-political entities
MALE	Person: male first name	**LOC**	Locations other than GPEs
LAST	Person: last name	**DAT**	Reference to a date or period
TTL	Honorific title	**LANG**	Any named language
NORP	Nationality, Religion, Political(adjectival)		

Table 3: Target categories

Corpus: In our experiments, we used the Google Web 1T corpus (Brants and Franz, 2006) as our expansion corpus. Specifically, we use the open source package *LIT-Indexer* (Ceylan and Mihalcea, 2011) to support efficient wildcard querying for pattern generation and instance extraction.

Target Expansion Categories: We conduct our experiments on thirteen categories, which are shown in Table 3. Eleven of them are from Curran et al. (2007). Besides the eleven categories, to evaluate how well ESE systems can resolve the semantic drift problem, we use two additional categories (*Capital* and *Chemical Element*) which are high likely to drift into other categories.

Evaluation Criteria: Following Curran et al (2007), we use *precision at top n (P@N)* as the performance metrics, i.e., the percentage of correct entities in the top n ranked entities for a given category. In our experiments, we use *P@10*, *P@20*, *P@50*, *P@100* and *P@200*. Since the output is a ranked list of extracted entities, we also choose the average precision (AP) as the evaluation metric. In our experiments, the correctness of all extracted entities is manually judged. In our experiments, we present results to 3 annotators, and an instance will be considered as positive if 2 annotators label it as positive. We also provide annotators some supporting resources for better evaluation, e.g., the entity list of target type collected from Wikipedia.

4.2 Experimental Results

In this section, we analyze the effect of negative instances, categories boundaries, and seed selection strategies. We compare our method with the following two baseline methods: i) ***Only_Pos (POS)***: This is an entity set expansion system which uses only positive seeds. ii) ***Mutual_Exclusion (ME)***: This is a mutual exclusion bootstrapping based ESE method, whose expansion boundary is determined by the exclusion of the categories.

We implement our method using two different settings: i) ***Hum_Co-Bootstrapping (Hum_CB)***: This is the proposed Co-Bootstrapping method in which the initial negative seeds are manually given. Specifically, we randomly select five positive seeds from the list of the category's instances while the initial negative seeds are manually provided. ii) ***Feedback_Co-Bootstrapping (FB_CB)***: This is our proposed probabilistic Co-Bootstrapping method with two steps of selecting initial negative seeds: 1) Expand the entity set using only the positive seeds for only first iteration. Return the top ten instances. 2) Select the negative instances in the top ten results of the first iteration as negative seeds.

4.2.1. Overall Performance

Several papers have shown that the experimental performance may vary with different seed choices (Kozareva and Hovy, 2010; McIntosh and Curran, 2009; Vvas et al., 2009). Therefore, we input the ESE system with five different positive seed settings for each category. Finally we average the performance on the five settings so that the impact of seed selection can be reduced.

	P@10	**P@20**	**P@50**	**P@100**	**P@200**	**MAP**
POS	0.84	0.74	0.55	0.41	0.34	0.42
ME	0.83(0.90)	0.79(0.87)	0.68(0.78)	0.58(0.67)	0.51(0.59)	-
Hum_CB	0.97	0.95	0.83	0.71	0.57	0.78
FB_CB	**0.97**	**0.96**	**0.90**	**0.79**	**0.66**	**0.85**

Table 4: The overall experimental results

Table 4 shows the overall experimental results. The results in parentheses are the known results of eleven categories (without *CAP* and *ELE*) shown in (Curran et al., 2007). MAP of ME is missed because there are no available results in (Curran et al., 2007). From Table 4, we can see that:

1) Our method can achieve a significant performance improvement: Compared with the baseline POS, our method Hum_CB and FB_CB can respectively achieve a 23% and 32% improvement on P@200; Compared with the baseline ME, our method Hum_CB and FB_CB can respectively improve P@200 by 6% and 15%.

2) By explicitly representing the expansion boundary, the expansion performance can be increased: Compared with the baseline POS, ME can achieve a 17% improvement on P@200, and our method Hum_CB can achieve a 23% improvement on P@200.

3) The negative seeds can better determine the expansion boundary than mutually exclusive categories. Compared with ME, Hum_CB and FB_CB can respectively achieve a 6% and 15% improvement on P@200. We believe this is because using negative instances is a more accurate and more robust way for defining and maintaining the expansion boundary than mutually exclusive categories.

4) The system’s feedback is useful for selecting negative instances: Compared with Hum_CB, FB_CB method can significantly improve the P@200 by 9.0%. We believe this is because that the system’s feedback is a good indicator of the semantic drift direction. In contrast, it is usually difficult for human to determine which directions the bootstrapping will drift towards.

4.2.2. Detailed Analysis: Expansion Boundary

In Table 5, we show the top 20 positive and negative *Capital* instances (FB_CB setting). From Table 5, we can make the following observations: 1) Our method can effectively generate negative instances. In Table 5, the negative instances contain cities, states, countries and general terms, all of which have a high semantic overlap with *Capital* category. 2) The positive instances and negative instances generated by our Co-Bootstrapping method can discriminately determine the expansion boundary. For instance, the negative instances *Kyoto* can distinguish *Capital* from *City*; *Australia* and *China* can distinguish *Capital* from *Country*;

Positive Instances		***London***, ***Paris***, ***Moscow***, ***Beijing***, ***Madrid***, ***Amsterdam***, ***Washington***, ***Tokyo***, ***Berlin***, ***Rome***, ***Vienna***, ***Baghdad***, ***Athens***, ***Bangkok***, ***Cairo***, ***Dublin***, ***Brussels***, ***Prague***, *San*, ***Budapest***
Negative Instances (with categories)	City	*Kyoto*, *Kong*, *Newcastle*, *Zurich*, *Lincoln*, *Albany*, *Lyon*, *LA*, *Shanghai*
	Country	*China*, *Australia*
	General	*downtown*, *April*
	State	*Hawaii*, *Oklahoma*, *Manhattan*
	Other	*Hollywood*, *DC*, ***Tehran***, ***Charlotte***

Table 5: Top 20 positive instances and negative instances (True positive instances are in bold)

4.2.3. Detailed Analysis: Semantic Drift Problem

POS	***Stockholm***, *Tampa*, *East*, *West*, *Springfield*, *Newport*, *Cincinnati*, ***Dublin***, *Chattanooga*, *Savannah*, *Omaha*, *Cambridge*, *Memphis*, *Providence*, ***Panama***, *Miami*, ***Cape***, *Victoria*, *Milan*, ***Berlin***
ME	***London***, ***Prague***, *Newport*, ***Cape***, ***Dublin***, *Savannah*, *Chattanooga*, ***Beijing***, *Memphis*, ***Athens***, ***Berlin***, *Miami*, ***Plymouth***, *Victoria*, *Omaha*, ***Tokyo***, *Portland*, *Troy*, *Anchorage*, ***Bangkok***
Hum_CB	***London***, ***Rome***, ***Berlin***, ***Paris***, ***Athens***, ***Moscow***, ***Tokyo***, ***Beijing***, ***Prague***, ***Madrid***, ***Vienna***, ***Dublin***, ***Budapest***, ***Amsterdam***, ***Bangkok***, ***Brussels***, *Sydney*, ***Cairo***, ***Washington***, *Barcelona*
FB_CB	***London***, ***Paris***, ***Moscow***, ***Beijing***, ***Washington***, ***Tokyo***, ***Berlin***, ***Rome***, ***Vienna***, ***Baghdad***, ***Athens***, ***Bangkok***, ***Cairo***, ***Brussels***, ***Prague***, *San*, ***Budapest***, ***Amsterdam***, ***Dublin***, ***Madrid***

Table 6: Top 20 instances of all methods (True positive instances are in bold)

To analyze how our method can resolve the semantic drift problem, Table 6 shows the top 20 positive *Capital* instances of different methods. From Table 6, we can make the following observations: i) Different methods can resolve the semantic drift problem to different extent: ME is better than POS, with 50% instances being positive, and our method is better than ME, with 95% instances being positive. ii) The Co-Bootstrapping method can effectively resolve the semantic drift problem: 25% of POS’s top 20 instances and 50% of ME’s top 20 instances are positive. In contrast, 90% of Hum_CB’s top 20 instances and 95% of FB_CB’s top 20 instances are positive respectively. It proves that Co-Bootstrapping method can better resolve the semantic drift problem than POS and ME.

4.3 Parameter Optimization

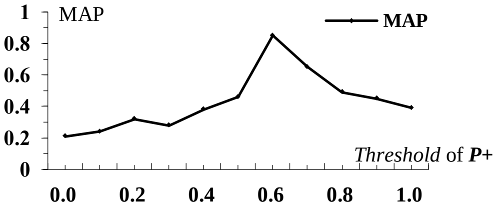

Figure 4: The MAP vs. *threshold* of ***P+***

Our method has only one parameter: *threshold* of ***P+***, which determines the instance’s polarity. Intuitively, a larger *threshold* of ***P+*** will improve the precision of the positive instances but will regard some positive instances as negative instances mistakenly. As shown in Figure 4, our method can achieve the best *MAP* performance when the value of the *threshold* is 0.6.

4.4 Comparison with State-of-the-Art Systems

We also compare our method with three state-of-the-art systems: *Google Sets*[1]-- an ESE application provided by Google, *SEAL*[2] -- a state-of-the-art ESE method proposed by Wang and Cohen (2008), and *WMEB* -- a state-of-the-art mutual exclusion based system proposed in McIntosh and Curran (2008). To make a fair comparison, we directly use the results before the adjustment which miss P@10 and P@50 in their original paper (McIntosh and Curran, 2008) and compared the performance of these systems on nine categories in (McIntosh and Curran, 2008). For each system, we conduct the experiment five times to reduce the impact of seeds selection. The average P@10, P@50, P@100 and P@200 are shown in Figure 5.

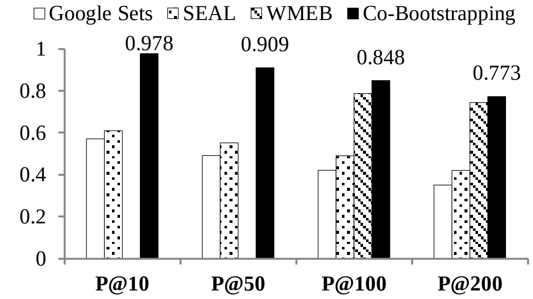

Figure 5: The results compared with three state-of-the-art systems

From the results shown in Figure 5, we can see that our probabilistic Co-Bootstrapping method can achieve state-of-the-art performance on all metrics: Compared with the well-known baseline *Google Sets*, our method can get a 42.0% improvement on P@200; Compared with the *SEAL* baseline, our method can get a 35.0% improvement on P@200; Compared with the WMEB method, our method can achieve a 6.2% improvement on P@100 and a 3.1% improvement on P@200.

5 Conclusion and Future Work

In this paper, we proposed a probabilistic Co-Bootstrapping method for entity set expansion. By introducing negative instances to define and refine the expansion boundary, our method can effectively resolve the expansion boundary problem and the semantic drift problem. Experimental results show that our method achieves significant performance improvement over the baselines, and outperforms three state-of-the-art ESE systems. Currently, our method did not take into account the long tail entity expansion, i.e., the instances which appear only a few times in the corpus, such as *Saipan, Roseau* and *Suva* for the *Capital* category. For future work, we will resolve the long tail entities in our Co-Bootstrapping method by taking the sparsity of instances/patterns into consideration.

6 Acknowledgements

We would like to thank three anonymous reviewers for invaluable comments and suggestions to improve our paper. This work is supported by the National Natural Science Foundation of China under Grants no. 61100152 and 61272324, and the National High Technology Development 863 Program of China under Grants no. 2013AA01A603.

References

Eugene Agichtein and Luis Gravano. 2000. *Snowball: Extracting Relations from Large Plain-Text Collections.* In: Proceedings of the fifth ACM conference on Digital libraries (DL-00), Pages 85-94.

Joohui An, Seungwoo Lee, and Gary Geunbae Lee. 2003. *Automatic acquisition of named entity tagged corpus from world wide web*. In: Proceedings of ACL-03, Pages 165-168, Volume 2.

Thorsten Brants and Alex Franz. 2006. *Web 1t-5gram version1*. http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2006T13

[1] https://docs.google.com/spreadsheet/

[2] http://www.boowa.com/

Sergey Brin. 1998. *Extracting patterns and relations from the World Wide Web*. In: Proceedings of the Workshop at the 6th International Conference on Extending Database Technology, Pages 172-183.

Michael J. Cafarella, Doug Downey, Stephen Soderland, and Oren Etzioni. 2005. *KnowItNow: Fast, Scalable Information Extraction from the Web*. In: Proceedings of EMNLP-05, Pages 563-570.

Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. 2008. *Context-aware query suggestion by mining click-through and session data*. In Proceedings of KDD-08, pages 875–883.

Hakan Ceylan and Rada Mihalcea. 2011. *An Efficient Indexer for Large N-Gram Corpora*. In: Proceedings of System Demonstrations of ACL-11, Pages 103-108.

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 KDD-04, Pages 89-98.

Alessandro Cucchiarelli and Paola Velardi. 2001. *Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence*. In: Computational Linguistics, Pages 123-131, Volume 27.

James R. Curran, Tara Murphy, and Bernhard Scholz. 2007. *Minimising semantic drift with Mutual Exclusion Bootstrapping*. In: Proceedings of the 10th Conference of the Pacific Association for Computational Linguistics, Pages 172–180.

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*. In: Artificial Intelligence, Pages 91-134, Volume 165.

Jian Hu, Gang Wang, Fred Lochovsky, Jiantao Sun, and Zheng Chen. 2009. *Understanding user's query intent with Wikipedia*. In Proceedings of WWW-09, Pages 471–480.

Zornitsa Kozareva and Eduard Hovy. 2010. *Learning arguments and supertypes of semantic relations using recursive patterns*. In: Proceedings of ACL-10, Pages 1482–1491.

Tara McIntosh and James R. Curran. 2008. *Weighted mutual exclusion bootstrapping for domain independent lexicon and template acquisition*. In: Proceedings of the Australasian Language Technology Association Workshop, Pages 97-105.

Tara McIntosh and James R. Curran. 2009. *Reducing semantic drift with bagging and distributional similarity*. In: Proceedings of ACL-09, Pages 396-404.

Patrick Pantel and Dekang Lin. 2002. *Discovering word senses from text*. In: Proceedings of KDD-08, Pages 613-619.

Patrick Pantel and Deepak Ravichandran. 2004. *Automatically Labeling Semantic Classes*. In: Proceedings of HLT/NAACL, Pages 321-328, Volume 4.

Patrick Pantel and Marco Pennacchiotti. 2006. *Espresso: Leveraging Generic Patterns for Automatically Harvesting Semantic Relations*. In: Proceedings of ACL-06, Pages 113–120.

Patrick Pantel, Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu and Vishnu Vyas. 2009. *Web-Scale Distributional Similarity and Entity Set Expansion*. In: Proceedings of EMNLP-09, Pages 938-947.

Marius Pasca. 2007. *Weakly-supervised discovery of named entities using web search queries*. In: Proceedings of CIKM-07, Pages 683-690.

Marco Pennacchiotti, Patrick Pantel. 2011. *Automatically building training examples for entity extraction*. In: Proceedings of CoNLL-11, Pages 163-171.

Ellen Riloff and Rosie Jones. 1999. *Learning dictionaries for information extraction using multi-level bootstrapping*. In: Proceedings of AAAI-99, Pages 474-479.

Partha P. Talukdar, Joseph Reisinger, Marius Pasca, Deepak Ravichandran, Rahul Bhagat, and Fernando Pereira. 2008. *Weakly-supervised acquisition of labeled class instances using graph random walks*. In: Proceedings of EMNLP-08, Pages 582-590.

Michael Thelen and Ellen Riloff. 2002. *A bootstrapping method for learning semantic lexicons using extraction pattern contexts*. In: Proceedings of ACL-02, Pages 214-221.

Richard C. Wang and William W. Cohen. 2008. *Iterative Set Expansion of Named Entities using the Web*. In: Proceedings of ICDM-08, Pages 1091-1096.

Richard C. Wang and William W. Cohen. 2009. *Automatic Set Instance Extraction using the Web*. In: Proceedings of ACL-09, Pages 441-449.

Vishnu Vvas, Patrick Pantel and Eric Crestan. 2009. *Helping editors choose better seed sets for entity set expansion*. In: Proceedings of CIKM-09, Pages 225-234

Roman Yangarber, Winston Lin and Ralph Grishman. 2002. *Unsupervised learning of generalized names*. In: Proceedings of COLING-02, Pages 1-7.