Distributional Similarity vs. PU Learning for Entity Set Expansion

Xiao-Li Li
Institute for Infocomm Research,
1 Fusionopolis Way #21-01 Connexis
Singapore 138632
xlli@i2r.a-star.edu.sg

Lei Zhang
University of Illinois at Chicago,
851 South Morgan Street, Chicago,
Chicago, IL 60607-7053, USA
zhang3@cs.uic.edu

Bing Liu
University of Illinois at Chicago,
851 South Morgan Street, Chicago,
Chicago, IL 60607-7053, USA
liub@cs.uic.edu

See-Kiong Ng
Institute for Infocomm Research,
1 Fusionopolis Way #21-01 Connexis
Singapore 138632
skng@i2r.a-star.edu.sg

Abstract

Distributional similarity is a classic technique for entity set expansion, where the system is given a set of seed entities of a particular class, and is asked to expand the set using a corpus to obtain more entities of the same class as represented by the seeds. This paper shows that a machine learning model called *positive and unlabeled learning* (PU learning) can model the set expansion problem better. Based on the test results of 10 corpora, we show that a PU learning technique outperformed distributional similarity significantly.

1 Introduction

The entity set expansion problem is defined as follows: Given a set *S* of seed entities of a particular class, and a set *D* of candidate entities (e.g., extracted from a text corpus), we wish to determine which of the entities in *D* belong to *S*. In other words, we "expand" the set *S* based on the given seeds. This is clearly a classification problem which requires arriving at a binary decision for each entity in *D* (belonging to *S* or not). However, in practice, the problem is often solved as a ranking problem, i.e., ranking the entities in *D* based on their likelihoods of belonging to *S*.

The classic method for solving this problem is based on *distributional similarity* (Pantel *et al.* 2009; Lee, 1998). The approach works by comparing the similarity of the surrounding word distributions of each candidate entity with the seed entities, and then ranking the candidate entities using their similarity scores.

In machine learning, there is a class of semi-supervised learning algorithms that learns from *positive* and *unlabeled* examples (PU learning for short). The key characteristic of PU learning is that there is no negative training example available for learning. This class of algorithms is less known to the natural language processing (NLP) community compared to some other semi-supervised learning models and algorithms.

PU learning is a two-class classification model. It is stated as follows (Liu *et al.* 2002): Given a set *P* of positive examples of a particular class and a set *U* of unlabeled examples (containing hidden positive and negative cases), a classifier is built using *P* and *U* for classifying the data in *U* or future test cases. The results can be either binary decisions (whether each test case belongs to the positive class or not), or a ranking based on how likely each test case belongs to the positive class represented by *P*. Clearly, the set expansion problem can be mapped into PU learning exactly, with *S* and *D* as *P* and *U* respectively.

This paper shows that a PU learning method called S-EM (Liu *et al.* 2002) outperforms distributional similarity considerably based on the results from 10 corpora. The experiments involved extracting named entities (e.g., product and organization names) of the same type or class as the given seeds. Additionally, we also compared S-EM with a recent method, called *Bayesian Sets* (Ghahramani and Heller, 2005), which was designed specifically for set expansion. It also does not perform as well as PU learning. We will explain why PU learning performs better than both methods in Section 5. We believe that this finding is of interest to the NLP community.

There is another approach used in the Web environment for entity set expansion. It exploits Web page structures to identify lists of items using wrapper induction or other techniques. The idea is that items in the same list are often of the same type. This approach is used by Google Sets (Google, 2008) and Boo!Wa! (Wang and Cohen, 2008). However, as it relies on Web page structures, it is not applicable to general free texts.

2 Three Different Techniques

2.1 Distributional Similarity

Distributional similarity is a classic technique for the entity set expansion problem. It is based on the hypothesis that words with similar meanings tend to appear in similar contexts (Harris, 1985). As such, a method based on distributional similarity typically fetches the surrounding contexts for each term (i.e. both seeds and candidates) and represents them as vectors by using TF-IDF or PMI (*Pointwise Mutual Information*) values (Lin, 1998; Gorman and Curran, 2006; Paşca *et al*. 2006; Agirre *et al*. 2009; Pantel *et al*. 2009). Similarity measures such as *Cosine*, *Jaccard*, *Dice*, etc, can then be employed to compute the similarities between each candidate vector and the seeds centroid vector (one centroid vector for all seeds). Lee (1998) surveyed and discussed various distribution similarity measures.

2.2 PU Learning and S-EM

PU learning is a semi-supervised or partially supervised learning model. It learns from positive and unlabeled examples as opposed to the model of learning from a small set of labeled examples of every class and a large set of unlabeled examples, which we call LU learning (L and U stand for *labeled* and *unlabeled* respectively) (Blum and Mitchell, 1998; Nigam *et al*. 2000)

There are several PU learning algorithms (Liu *et al*. 2002; Yu *et al*. 2002; Lee and Liu, 2003; Li *et al*. 2003; Elkan and Noto, 2008). In this work, we used the S-EM algorithm given in (Liu *et al*. 2002). S-EM is efficient as it is based on naïve Bayesian (NB) classification and also performs well. The main idea of S-EM is to use a *spy* technique to identify some *reliable negatives* (*RN*) from the unlabeled set *U*, and then use an EM algorithm to learn from *P*, *RN* and *U–RN*.

The *spy* technique in S-EM works as follows (Figure 1): First, a small set of positive examples (denoted by *SP*) from *P* is randomly sampled (line 2). The default sampling ratio in S-EM is $s = 15\%$, which we also used in our experiments. The positive examples in *SP* are called "spies". Then, a NB classifier is built using the set *P– SP* as positive and the set $U \cup SP$ as negative (line 3, 4, and 5). The NB classifier is applied to classify each $u \in U \cup SP$, i.e., to assign a probabilistic class label $p(+|u)$ (+ means positive). The probabilistic labels of the spies are then used to decide reliable negatives (*RN*). In particular, a probability threshold *t* is determined using the probabilistic labels of spies in *SP* and the input parameter *l* (noise level). Due to space constraints, we are unable to explain *l*. Details can be found in (Liu *et al*. 2002). *t* is then used to find *RN* from *U* (lines 8-10). The idea of the spy technique is clear. Since spy examples are from *P* and are put into *U* in building the NB classifier, they should behave similarly to the hidden positive cases in *U*. Thus, they can help us find the set *RN*.

```
Algorithm Spy($P$, $U$, $s$, $l$)
1. $RN \leftarrow \emptyset$;                // Reliable negative set
2. $SP \leftarrow Sample(P, s\%)$;
3. Assign each example in $P - SP$ the class label +1;
4. Assign each example in $U \cup SP$ the class label -1;
5. $C \leftarrow NB(P - S, U \cup SP)$; // Produce a NB classifier
6. Classify each $u \in U \cup SP$ using $C$;
7. Decide a probability threshold $t$ using $SP$ and $l$;
8. for each $u \in U$ do
9.     if its probability $p(+|u) < t$ then
10.        $RN \leftarrow RN \cup \{u\}$;
```

Figure 1. Spy technique for extracting reliable negatives (*RN*) from *U*.

Given the positive set *P*, the reliable negative set *RN* and the remaining unlabeled set *U–RN*, an Expectation-Maximization (EM) algorithm is run. In S-EM, EM uses the naïve Bayesian classification as its base method. The detailed algorithm is given in (Liu *et al*. 2002).

2.3 Bayesian Sets

Bayesian Sets, as its name suggests, is based on Bayesian inference, and was designed specifically for the set expansion problem (Ghahramani and Heller, 2005). The algorithm learns from a seeds set (i.e., a positive set *P*) and an unlabeled candidate set *U*. Although it was not designed as a PU learning method, it has similar characteristics and produces similar results as PU learning. However, there is a major difference. PU learning is a classification model, while Bayesian Sets is a ranking method. This difference has a major implication on the results that they produce as we will discuss in Section 5.3.

In essence, Bayesian Sets learns a score function using P and U to generate a score for each unlabeled case $u \in U$. The function is as follows:

$$score(u) = \frac{p(u \mid P)}{p(u)} \quad (1)$$

where $p(u|P)$ represents how probable u belongs to the positive class represented by P. $p(u)$ is the prior probability of u. Using the Bayes' rule, equation (1) can be re-written as:

$$score(u) = \frac{p(u, P)}{p(u)p(P)} \quad (2)$$

Following the idea, Ghahramani and Heller (2005) proposed a computable score function. The scores can be used to rank the unlabeled candidates in U to reflect how likely each $u \in U$ belongs to P. The mathematics for computing the score is involved. Due to the limited space, we cannot discuss it here. See (Ghahramani and Heller, 2005) for details. In (Heller and Ghahramani, 2006), Bayesian Sets was also applied to an image retrieval application.

3 Data Generation for Distributional Similarity, Bayesian Sets and S-EM

Preparing the data for distributional similarity is fairly straightforward. Given the seeds set S, a seeds centroid vector is produced using the surrounding word contexts (see below) of all occurrences of all the seeds in the corpus (Pantel et al, 2009). In a similar way, a centroid is also produced for each candidate (or unlabeled) entity.

Candidate entities: Since we are interested in named entities, we select single words or phrases as candidate entities based on their corresponding part-of-speech (POS) tags. In particular, we choose the following POS tags as entity indicators — NNP (proper noun), NNPS (plural proper noun), and CD (cardinal number). We regard a phrase (could be one word) with a sequence of NNP, NNPS and CD POS tags as one candidate entity (CD cannot be the first word unless it starts with a letter), e.g., "Windows/NNP 7/CD" and "Nokia/NNP N97/CD" are regarded as two candidates "Windows 7" and "Nokia N97".

Context: For each seed or candidate occurrence, the *context* is its set of surrounding words within a window of size w, i.e. we use w words right before the seed or the candidate and w words right after it. Stop words are removed.

For S-EM and Bayesian Sets, both the positive set P (based on the seeds set S) and the unlabeled candidate set U are generated differently. They are not represented as centroids.

Positive and unlabeled sets: For each seed $s_i \in S$, each occurrence in the corpus forms a vector as a positive example in P. The vector is formed based on the surrounding words context (see above) of the seed mention. Similarly, for each candidate $d \in D$ (see above; D denotes the set of all candidates), each occurrence also forms a vector as an unlabeled example in U. Thus, each unique seed or candidate entity may produce multiple feature vectors, depending on the number of times that it appears in the corpus.

The components in the feature vectors are term frequencies for S-EM as S-EM uses naïve Bayesian classification as its base classifier. For Bayesian Sets, they are 1's and 0's as Bayesian Sets only takes binary vectors based on whether a term occurs in the context or not.

4 Candidate Ranking

For distributional similarity, ranking is done using the similarity value of each candidate's centroid and the seeds' centroid (one centroid vector for all seeds). Rankings for S-EM and Bayesian Sets are more involved. We discuss them below.

After it ends, S-EM produces a Bayesian classifier C, which is used to classify each vector $u \in U$ and to assign a probability $p(+|u)$ to indicate the likelihood that u belongs to the positive class. Similarly, Bayesian Sets produces a score $score(u)$ for each u (not a probability).

Recall that for both S-EM and Bayesian Sets, each unique candidate entity may generate multiple feature vectors, depending on the number of times that the candidate entity occurs in the corpus. As such, the rankings produced by S-EM and Bayesian Sets are not the rankings of the entities, but rather the rankings of the entities' occurrences. Since different vectors representing the same candidate entity can have very different probabilities (for S-EM) or scores (for Bayesian Sets), we need to combine them and compute a single score for each unique candidate entity for ranking.

To this end, we also take the entity frequency into consideration. Typically, it is highly desirable to rank those correct and frequent entities at the top because they are more important than the infrequent ones in applications. With this in mind, we define a ranking method.

Let the probabilities (or scores) of a candidate entity $d \in D$ be $V_d = \{v_1, v_2 \ldots, v_n\}$ for the n feature vectors of the candidate. Let M_d be the median of V_d. The final score (fs) for d is defined as:

$$fs(d) = M_d \times \log(1+n) \quad (3)$$

The use of the median of V_d can be justified based on the statistical *skewness* (Neter *et al.* 1993). If the values in V_d are skewed towards the high side (negative skew), it means that the candidate entity is very likely to be a true entity, and we should take the median as it is also high (higher than the mean). However, if the skew is towards the low side (positive skew), it means that the candidate entity is unlikely to be a true entity and we should again use the median as it is low (lower than the mean) under this condition.

Note that here n is the frequency count of candidate entity d in the corpus. The constant 1 is added to smooth the value. The idea is to push the frequent candidate entities up by multiplying the logarithm of frequency. *log* is taken in order to reduce the effect of big frequency counts.

The final score $fs(d)$ indicates candidate d's overall likelihood to be a relevant entity. A high $fs(d)$ implies a high likelihood that d is in the expanded entity set. We can then rank all the candidates based on their $fs(d)$ values.

5 Experimental Evaluation

We empirically evaluate the three techniques in this section. We implemented *distribution similarity* and *Bayesian Sets*. S-EM was downloaded from http://www.cs.uic.edu/~liub/S-EM/S-EM-download.html. For both Bayesian Sets and S-EM, we used their default parameters. EM in S-EM ran only two iterations. For distributional similarity, we tested TF-IDF and PMI as feature values of vectors, and Cosine and Jaccard as similarity measures. Due to space limitations, we only show the results of the PMI and Cosine combination as it performed the best. This combination was also used in (Pantel *et al.*, 2009).

5.1 Corpora and Evaluation Metrics

We used 10 diverse corpora to evaluate the techniques. They were obtained from a commercial company. The data were crawled and extracted from multiple online message boards and blogs discussing different products and services. We split each message into sentences, and the sentences were POS-tagged using Brill's tagger (Brill, 1995). The tagged sentences were used to extract candidate entities and their contexts. Table 1 shows the domains and the number of sentences in each corpus, as well as the three seed entities used in our experiments for each corpus. The three seeds for each corpus were randomly selected from a set of common entities in the application domain.

Table 1. Descriptions of the 10 corpora

Domains	# Sentences	Seed Entities
Bank	17394	Citi, Chase, Wesabe
Blu-ray	7093	S300, Sony, Samsung
Car	2095	Honda, A3, Toyota
Drug	1504	Enbrel, Hurmia, Methotrexate
Insurance	12419	Cobra, Cigna, Kaiser
LCD	1733	PZ77U, Samsung, Sony
Mattress	13191	Simmons, Serta, Heavenly
Phone	14884	Motorola, Nokia, N95
Stove	25060	Kenmore, Frigidaire, GE
Vacuum	13491	Dc17, Hoover, Roomba

The regular evaluation metrics for named entity recognition such as precision and recall are not suitable for our purpose as we do not have the complete sets of gold standard entities to compare with. We adopt rank precision, which is commonly used for evaluation of entity set expansion techniques (Pantel *et al.*, 2009):

Precision @ N: The percentage of correct entities among the top *N* entities in the ranked list.

5.2 Experimental Results

The detailed experimental results for window size 3 (w=3) are shown in Table 2 for the 10 corpora. We present the precisions at the top 15-, 30- and 45-ranked positions (i.e., precisions @15, 30 and 45) for each corpus, with the average given in the last column. For distributional similarity, to save space Table 2 only shows the results of **Distr-Sim-freq**, which is the distributional similarity method with term frequency considered in the same way as for Bayesian Sets and S-EM, instead of the original distributional similarity, which is denoted by **Distr-Sim**. This is because on average, **Distr-Sim-freq** performs better than **Distr-Sim**. However, the summary results of both **Distr-Sim-freq** and **Distr-Sim** are given in Table 3.

From Table 2, we observe that on average **S-EM** outperforms **Distr-Sim-freq** by about 12 – 20% in terms of *Precision @ N*. **Bayesian-Sets** is also more accurate than **Distr-Sim-freq**, but **S-EM** outperforms **Bayesian-Sets** by 9 – 10%.

To test the sensitivity of window size w, we also experimented with w = 6 and w = 9. Due to space constraints, we present only their average results in Table 3. Again, we can see the same performance pattern as in Table 2 (w = 3): **S-EM** performs the best, **Bayesian-Sets** the second, and the two distributional similarity methods the third and the fourth, with **Distr-Sim-freq** slightly better than **Distr-Sim**.

Table 2. *Precision @ top N* (with 3 seeds, and window size *w* = 3)

	Bank	Blu-ray	Car	Drug	Insurance	LCD	Mattress	Phone	Stove	Vacuum	**Avg.**
	Top 15										
Distr-Sim-freq	0.466	0.333	0.800	0.666	0.666	0.400	0.666	0.533	0.666	0.733	**0.592**
Bayesian-Sets	0.533	0.266	0.600	0.666	0.600	0.733	0.666	0.533	0.800	0.800	**0.617**
S-EM	0.600	0.733	0.733	0.733	0.533	0.666	0.933	0.533	0.800	0.933	**0.720**
	Top 30										
Distr-Sim-freq	0.466	0.266	0.700	0.600	0.500	0.333	0.500	0.466	0.600	0.566	**0.499**
Bayesian-Sets	0.433	0.300	0.633	0.666	0.400	0.566	0.700	0.333	0.833	0.700	**0.556**
S-EM	0.500	0.700	0.666	0.666	0.566	0.566	0.733	0.600	0.600	0.833	**0.643**
	Top 45										
Distr-Sim-freq	0.377	0.288	0.555	0.500	0.377	0.355	0.444	0.400	0.533	0.400	**0.422**
Bayesian-Sets	0.377	0.333	0.666	0.555	0.377	0.511	0.644	0.355	0.733	0.600	**0.515**
S-EM	0.466	0.688	0.644	0.733	0.533	0.600	0.644	0.555	0.644	0.688	**0.620**

Table 3. Average precisions over the 10 corpora of different window size (3 seeds)

	Window-size *w* = 3				**Window-size *w* = 6**				**Window-size *w* = 9**		
Top Results	**Top 15**	**Top 30**	**Top 45**		**Top 15**	**Top 30**	**Top 45**		**Top 15**	**Top 30**	**Top 45**
Distr-Sim	0.579	0.466	0.410		0.553	0.483	0.439		0.519	0.473	0.412
Distr-Sim-freq	0.592	0.499	0.422		0.553	0.492	0.441		0.559	0.476	0.410
Bayesian-Sets	0.617	0.556	0.515		0.593	0.539	0.524		0.539	0.522	0.497
S-EM	**0.720**	**0.643**	**0.620**		**0.666**	**0.606**	**0.597**		**0.666**	**0.620**	**0.604**

5.3 Why does S-EM Perform Better?

From the tables, we can see that both S-EM and Bayesian Sets performed better than distributional similarity. S-EM is better than Bayesian Sets. We believe that the reason is as follows: Distributional similarity does not use any information in the candidate set (or the unlabeled set *U*). It tries to rank the candidates solely through similarity comparisons with the given seeds (or positive cases). Bayesian Sets is better because it considers *U*. Its learning method produces a weight vector for features based on their occurrence differences in the positive set *P* and the unlabeled set *U* (Ghahramani and Heller 2005). This weight vector is then used to compute the final scores used in ranking. In this way, Bayesian Sets is able to exploit the useful information in *U* that was ignored by distributional similarity. S-EM also considers these differences in its NB classification; in addition, it uses the reliable negative set (*RN*) to help distinguish negative and positive cases, which both Bayesian Sets and distributional similarity do not do. We believe this balanced attempt by S-EM to distinguish the positive and negative cases is the reason for the better performance of S-EM. This raises an interesting question. Since Bayesian Sets is a ranking method and S-EM is a classification method, can we say even for ranking (our evaluation is based on ranking) classification methods produce better results than ranking methods? Clearly, our single experiment cannot answer this question. But intuitively, classification, which separates positive and negative cases by pulling them towards two opposite directions, should perform better than ranking which only pulls the data in one direction. Further research on this issue is needed.

6 Conclusions and Future Work

Although distributional similarity is a classic technique for entity set expansion, this paper showed that PU learning performs considerably better on our diverse corpora. In addition, PU learning also outperforms Bayesian Sets (designed specifically for the task). In our future work, we plan to experiment with various other PU learning methods (Liu *et al.* 2003; Lee and Liu, 2003; Li *et al.* 2007; Elkan and Noto, 2008) on this entity set expansion task, as well as other tasks that were tackled using distributional similarity. In addition, we also plan to combine some syntactic patterns (Etzioni *et al.* 2005; Sarmento *et al.* 2007) to further improve the results.

Acknowledgements: Bing Liu and Lei Zhang acknowledge the support of HP Labs Innovation Research Grant 2009-1062-1-A, and would like to thank Suk Hwan Lim and Eamonn O'Brien-Strain for many helpful discussions.

References

Agirre, E., Alfonseca, E., Hall, K., Kravalova, J., Pasca, M., and Soroa, A. 2009. *A study on similarity and relatedness using distributional and WordNet-based approaches.* NAACL HLT.

Blum, A. and Mitchell, T. 1998. *Combining labeled and unlabeled data with co-training.* In Proc. of Computational Learning Theory, pp. 92–100, 1998.

Brill, E. 1995. *Transformation-Based error-Driven learning and natural language processing: a case study in part of speech tagging. Computational Linguistics*.

Bunescu, R. and Mooney, R. 2004. *Collective information extraction with relational Markov Networks.* ACL.

Cheng T., Yan X. and Chang C. K. 2007. *Entity-Rank: searching entities directly and holistically.* VLDB.

Chieu, H.L. and Ng, H. Tou. 2002. *Name entity recognition: a maximum entropy approach using global information.* In The 6th Workshop on Very Large Corpora.

Downey, D., Broadhead, M. and Etzioni, O. 2007. *Locating complex named entities in Web Text.* IJCAI.

Elkan, C. and Noto, K. 2008. *Learning classifiers from only positive and unlabeled data.* KDD, 213-220.

Etzioni, O., Cafarella, M., Downey. D., Popescu, A., Shaked, T., Soderland, S., Weld, D. Yates. 2005. A. *Unsupervised named-entity extraction from the Web: An Experimental Study. Artificial Intelligence,* 165(1):91-134.

Ghahramani, Z and Heller, K.A. 2005. *Bayesian sets.* NIPS.

Google Sets. 2008. *System and methods for automatically creating lists.* US Patent: US7350187, March 25.

Gorman, J. and Curran, J. R. 2006. *Scaling distributional similarity to large corpora.* ACL.

Harris, Z. Distributional Structure. 1985. In: Katz, J. J. (ed.), *The philosophy of linguistics.* Oxford University Press.

Heller, K. and Ghahramani, Z. 2006. *A simple Bayesian framework for content-based image retrieval.* CVPR.

Isozaki, H. and Kazawa, H. 2002. *Efficient support vector classifiers for named entity recognition.* COLING.

Jiang, J. and Zhai, C. 2006. *Exploiting domain structure for named entity recognition.* HLT-NAACL.

Lafferty J., McCallum A., and Pereira F. 2001. *Conditional random fields: probabilistic models for segmenting and labeling sequence data.* ICML.

Lee, L. 1999. *Measures of distributional similarity*. ACL.

Lee, W-S. and Liu, B. 2003. *Learning with Positive and Unlabeled Examples Using Weighted Logistic Regression*. ICML.

Li, X., Liu, B. 2003. *Learning to classify texts using positive and unlabeled data*, IJCAI.

Li, X., Liu, B., Ng, S. 2007. *Learning to identify unexpected instances in the test sSet*. IJCAI.

Lin, D. 1998. *Automatic retrieval and clustering of similar words.* COLING/ACL.

Liu, B, Lee, W-S, Yu, P. S, and Li, X. 2002. *Partially supervised text classification.* ICML, 387-394.

Liu, B, Dai, Y., Li, X., Lee, W-S., and Yu. P. 2003. *Building text classifiers using positive and unlabeled examples*. ICDM, 179-188.

Neter, J., Wasserman, W., and Whitmore, G. A. 1993. *Applied Statistics*. Allyn and Bacon.

Nigam, K., McCallum, A., Thrun, S. and Mitchell, T. 2000. *Text classification from labeled and unlabeled documents using EM.* Machine Learning, 39(2/3), 103–134.

Pantel, P., Eric Crestan, Arkady Borkovsky, Ana-Maria Popescu, Vishnu, Vyas. 2009. *Web-Scale Distributional similarity and entity set expansion,* EMNLP.

Paşca, M. Lin, D. Bigham, J. Lifchits, A. Jain, A. 2006. *Names and similarities on the web: fast extraction in the fast lane.* ACL.

Sarmento, L., Jijkuon, V. de Rijke, M. and Oliveira, E. 2007. *"More like these": growing entity classes from seeds.* CIKM.

Wang, R. C. and Cohen, W. W. 2008. *Iterative set expansion of named entities using the web.* ICDM.

Yu, H., Han, J., K. Chang. 2002. *PEBL: Positive example based learning for Web page classification using SVM*. KDD, 239-248.