Distributed Representations of Words to Guide Bootstrapped Entity Classifiers

Sonal Gupta
Department of Computer Science
Stanford University
sonal@cs.stanford.edu

Christopher D. Manning
Department of Computer Science
Stanford University
manning@stanford.edu

Abstract

Bootstrapped classifiers iteratively generalize from a few seed examples or prototypes to other examples of target labels. However, sparseness of language and limited supervision make the task difficult. We address this problem by using distributed vector representations of words to aid the generalization. We use the word vectors to expand entity sets used for training classifiers in a bootstrapped pattern-based entity extraction system. Our experiments show that the classifiers trained with the expanded sets perform better on entity extraction from four online forums, with 30% F_1 improvement on one forum. The results suggest that distributed representations can provide good directions for generalization in a bootstrapping system.

1 Introduction

Bootstrapped or distantly-supervised learning is a form of semi-supervised learning, in which supervision is provided by seed examples. Supervised machine learning systems, on the other hand, require hand-labeling sufficient data to train a model, which can be costly and time consuming. Bootstrapped information extraction (IE) has become even more pertinent with the ever-growing amount of data coupled with the emergence of open IE systems (Carlson et al., 2010; Fader et al., 2011) and shared tasks like TAC-KBP.[1]

Limited supervision provided in bootstrapped systems, though an attractive quality, is also one of its main challenges. When seed sets are small, noisy, or do not cover the label space, the bootstrapped classifiers do not generalize well.

We use a major guiding inspiration of deep learning: we can learn a lot about syntactic and semantic similarities between words in an unsupervised fashion and capture this information in word vectors. This distributed representation can inform an inductive bias to generalize in a bootstrapping system.

In this paper, we present a simple approach of using the distributed vector representations of words to expand training data for entity classifiers in a bootstrapped system (see Algorithm 1). To improve the step of learning an entity classifier, we first learn vector representation of entities using the continuous bag of words model (Mikolov et al., 2013a). We then use kNN to expand the training set of the classifier by adding unlabeled entities close to seed entities in the training set. The key insight is to use the word vector similarity indirectly by enhancing training data for the entity classifier. We do not directly label the unlabeled entities using the similarity between word vectors, which we show extracts many noisy entities. We show that classifiers trained with expanded sets of entities perform better on extracting drug-and-treatment entities from four online health forums from MedHelp.[2]

2 Related Work

Bootstrapping has many variants, such as self-training, co-training, and label propagation. Yarowsky's style of self-training algo-

[1] http://www.nist.gov/tac/2014/KBP

[2] http://www.medhelp.org

rithms (Yarowsky, 1995) have been shown to be successful at bootstrapping (Collins and Singer, 1999). Co-training (Blum and Mitchell, 1998) and its boostrapped adaptation (Collins and Singer, 1999) require disjoint views of the features of the data. Whitney and Sarkar (2012) proposed a modified Yarowsky algorithm that used label propagation on graphs, inspired by Subramanya et al. (2010) algorithm that used a large labeled data for domain adaptation.

In this paper, we use the setting of bootstrapped pattern-based entity extraction (Riloff, 1996; Thelen and Riloff, 2002). This can be viewed as a form of the Yarowsky algorithm, with pattern learning as an additional step. Pattern based approaches have been widely used for IE (Chiticariu et al., 2013; Fader et al., 2011; Etzioni et al., 2005). Patterns are useful in two ways: they are good features, and they identify promising candidate entities. Recently, Gupta and Manning (2014) improved pattern scoring (Step 2 in Algorithm 1) using predicted labels of unlabeled entities. For entity scoring (Step 3), they used an average of feature values to predict the scores. We use the same framework but focus on improving the entity classifiers.

In most IE systems, including ours, word classes or word vectors are used as features in a classifier (Haghighi and Klein, 2006; Ratinov and Roth, 2009).

To the best of our knowledge, our work is the first to use distributed representations of words to improve a bootstrapped system by expanding the training set.

3 Background

In a bootstrapped pattern-based entity learning system, seed dictionaries and/or patterns provide weak supervision to label data. The system iteratively learns new entities belonging to a specific label from unlabeled text (Riloff, 1996; Collins and Singer, 1999) using patterns, such as lexico-syntactic surface word patterns (Hearst, 1992) and dependency tree patterns (Yangarber et al., 2000). We use lexico-syntactic surface word patterns to extract entities from unlabeled text starting with seed dictionaries for multiple classes. Algorithm 1 gives an overview. In this paper, we focus on improving the entity classifier (Step 3) by expanding its training data using distributed vector representations of words.

Algorithm 1 Bootstrapped Pattern-based Entity Extraction

```
Given: Text $D$, labels $L$, seed entities $E_l$ $\forall l \in$ L
while not-terminating-condition (e.g. precision is high) do
  for $l \in L$ do
    1. Label $D$ with $E_l$
    2. Create patterns around labeled entities.
       Learn good patterns and use them to extract candidate entities $C_l$.
    3. Learn an entity classifier and classify $C_l$.
       Add new classified entities to $E_l$.
```

Labeling known entities: The text is labeled using the label dictionaries, starting with the seed dictionaries in the first iteration.

Creating and Learning Patterns: Patterns are then created using the context around the labeled entities to create candidate patterns for label l. Candidate patterns are scored using a pattern scoring measure and the top ones are added to the list of learned patterns for label l. In our experiments, we use a widely used pattern scoring measure, RlogF (Riloff, 1996; Thelen and Riloff, 2002). Top ranked patterns with scores above a certain threshold are used to extract candidate entities C_l from text.

Learning entities: An entity classifier predicts the labels of C_l and adds the newly classified entities to label l's dictionary, E_l. We discard common words, negative entities, and those containing non-alphanumeric characters from the set.

Entity Classifier We build a one-vs-all entity classifier using logistic regression. In each iteration, for label l, the entity classifier is trained by treating l's dictionary entities (seed and learned in previous iterations) as positive and entities belonging to all other labels as negative. To improve generalization, we also sample the unlabeled entities that are not function words as negative. To train with a balanced dataset, we randomly sub-sample the negatives such that the number of negative instances is equal to the number of positive instances. The features for the entities are similar to Gupta and Manning (2014): edit distances from positive and negative entities, relative frequency of the entity words in the seed dictionaries, word classes computed using the Brown clustering algorithm (Brown et al., 1992; Liang, 2005), and pattern TF-IDF score. The last feature gives higher scores to entities that are extracted by many learned patterns and have low frequency in the dataset. In our experiments, we call this classifier as *NotExpanded*.

4 Approach

The lack of labeled data to train a good entity classifier is one of the challenges in bootstrapped learning. We use distributed representations of words, in the form of word vectors, to guide the entity classifier by expanding its training set. As explained in the previous section, we train a one-vs-all entity classifier in each iteration of the bootstrapped entity extraction for each label. We use unlabeled entities that are similar to the seed entities of the label as positive examples, and use unlabeled entities that are similar to seed entities of other labels as negative examples.[3]

To compute similarity of an unlabeled entity to the positive entities, we find k most similar positive entities, measured by cosine similarity between the word vectors, and average the scores. Similarly, we compute similarity of the unlabeled entity to the negative entities. If the entity's positive similarity score is above a given threshold θ and is higher than its negative similarity score, it is added to the training set with positive label. We expand the negative entities similarly.[4]

An alternative to our approach is to directly label the entities using the vector similarities. Our experimental results suggest that even though exploiting similarities between word vectors is useful for guiding the classifier by expanding the training set, it is not robust enough to use for labeling entities directly. For example, for our development dataset, when θ was set as 0.4, 16 out of 41 unlabeled entities that were expanded into the training set as positive entities were false positives.[5] Thus, labeling entities solely based on similarity scores resulted in lower performance. A classifier, on the other hand, can use other sources of information as features to predict an entity's label.

We compute the distributed vector representations using the continuous bag-of-words model (Mikolov et al., 2013a; Mikolov et al., 2013b) implemented in the word2vec toolkit.[6] We train 200-dimensional vector representations on a combined dataset of a 2014 Wikipedia dump (1.6 billion tokens), a sample of 50 million tweets from Twitter (200 million tokens), and an in-domain dataset of all MedHelp forums (400 million tokens). We removed words that occurred less than 20 times, resulting in a vocabulary of 89k words. We call this dataset Wiki+Twit+MedHelp. We used the parameters suggested in Pennington et al. (2014): negative sampling with 10 samples and a window size of 10. We ran the model for 3 iterations.

5 Experimental Setup

We present results on the same experimental setup, dataset, and seed lists as used in Gupta and Manning (2014). The task is to extract drug-and-treatment (DT) entities in sentences from four forums on the MedHelp user health discussion website: 1. Asthma, 2. Acne, 3. Adult Type II Diabetes (called Diabetes), and 4. Ear Nose & Throat (called ENT). A DT entity is defined as a pharmaceutical drug, or any treatment or intervention mentioned that may help a symptom or a condition. The output of all systems were judged by the authors, following the guidelines in (Gupta and Manning, 2014). We used Asthma as the development forum for parameter and threshold tuning. We used threshold θ as 0.4 and use k (number of nearest neighbors) as 2 when expanding the seed sets.

We evaluate systems by their precision and recall. Precision is defined as the fraction of correct entities among the entities extracted. Similar to (Gupta and Manning, 2014), we present the precision and recall curves for precision above 75% to compare systems when they extract entities with reasonably

[3]We take the cautious approach of finding similar entities only to the seed entities and not the learned entities. The algorithm can be modified to find similar entities to learned entities as well. Cautious approaches have been shown to be better for bootstrapped learning (Abney, 2004).

[4]We tried expanding just the positive entities and just the negative entities. Their relative performance, though higher than the baselines, varied between the datasets. Thus, for conciseness, we present results only for expanding both positives and negatives.

[5]Increasing θ extracted far fewer entities. $\theta = 0.5$ extracted only 5 entities, all true positives, and $\theta = 0.6$ extracted none.

[6]`http://code.google.com/p/word2vec/`

Forum	Expanded	Expanded-M	NotExpanded	Average
Asthma	**77.01**	75.68	74.48	65.42
Acne	73.84	**75.41**	71.65	65.05
Diabetes	**82.37**	44.25	48.75	21.82
ENT	**80.66**	80.04	77.02	59.50

Table 1: Area under Precision-Recall curve for all the systems. Expanded is our system when word vectors are learned using the Wiki+Twit+MedHelp data and Expanded-M is when word vectors are learning using the MedHelp data.

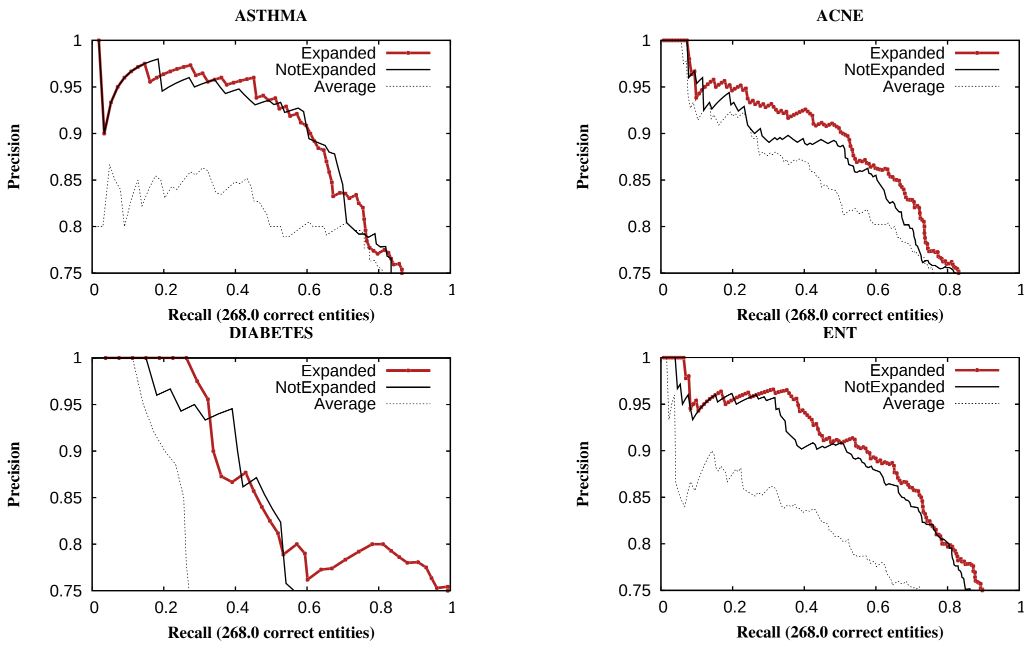

Figure 1: Precision vs. Recall curves of our system and the baselines for the four forums.

high precision. Recall is defined as the fraction of correct entities among the total unique correct entities pooled from all systems.[7] We calculate the area under the precision-recall curves (AUC-PR) to compare the systems.

We call our system *Expanded* in the experiments. To compare the effects of word vectors learned using different types of datasets, we also study our system when the word vectors are learned using just the in-domain MedHelp data, called *Expanded-M*. We compare against two baselines: *NotExpanded* as explained in Section 3, and *Average*, in which we average the feature values, similar to (Gupta and Manning, 2014).

[7]Note that calculating lower precisions or true recall is very hard to compute. Our dataset is unlabeled and manually labeling all entities is expensive. Pooling is a common evaluation strategy in such situations (such as, TAC-KBP shared task).

6 Results and Discussion

Table 1 shows AUC-PR of various systems and Figure 1 shows the precision-recall curves. Our systems *Expanded* and *Expanded-M*, which used similar entities for training, improved the scores for all four forums. We believe the improvement for the Diabetes forum was much higher than other forums because the baseline's performance on the forum degraded quickly in later iterations (see the figure), and improving the classifier helped in adding more correct entities. Additionally, Diabetes DT entities are more lifestyle-based and hence occur frequently in web text, making the word vectors trained using the Wiki+Twit+MedHelp dataset better suited.

In three out of four forums, word vectors trained

Positives	Negatives
Asthma	
pranayama, sterilizing, expectorants, inhalable, sanitizers, ayurvedic	block, yougurt, medcine, exertion, hate, virally
Diabetes	
quinoa, vinegars, vegatables, threadmill, possilbe, asanas, omegas	nicely, chiropracter, exhales, paralytic, metabolize, fluffy

Table 2: Examples of unlabeled entities that were expanded into the training sets. Gray colored entities were judged by the authors as falsely labeled.

using a large corpus perform better than those trained using the smaller in-domain corpus. For the Acne forum, where brand name DT entities are more frequent, the entities expanded by MedHelp vectors had fewer false positives than those expanded by Wiki+Twit+MedHelp.

Table 2 shows some examples of unlabeled entities that were included as positive/negative entities in the entity classifiers. Even though some entities were included in the training data with wrong labels, overall the classifiers benefited from the expansion.

7 Conclusion

We improve entity classifiers in bootstrapped entity extraction systems by enhancing the training set using unsupervised distributed representations of words. The classifiers learned using the expanded seed sets extract entities with better F_1 score. This supports our hypothesis that generalizing labels to entities that are similar according to unsupervised methods of word vector learning is effective in improving entity classifiers, notwithstanding that the label generalization is quite noisy.

References

S. Abney. 2004. Understanding the Yarowsky algorithm. *Computational Linguistics*, 30:365–395.

A. Blum and T. Mitchell. 1998. Combining labeled and unlabeled data with co-training. In *Conference on Learning Theory (COLT)*.

P. F. Brown, V. J. D. Pietra, P. V. deSouza, J. C. Lai, and R. L. Mercer. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18:467–479.

A. Carlson, J. Betteridge, R. C. Wang, Jr. E. R. Hruschka, and T. M. Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Web Search and Data Mining (WSDM)*, pages 101–110.

L. Chiticariu, Y. Li, and F. R. Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 827–832.

M. Collins and Y. Singer. 1999. Unsupervised models for named entity classification. In *Empirical Methods in Natural Language Processing (EMNLP)*.

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

A. Fader, S. Soderland, and O. Etzioni. 2011. Identifying relations for open information extraction. In *Empirical Methods in Natural Language Processing (EMNLP)*.

S. Gupta and C. D. Manning. 2014. Improved pattern learning for bootstrapped entity extraction. In *Computational Natural Language Learning (CoNLL)*.

A. Haghighi and D. Klein. 2006. Prototype-driven learning for sequence models. In *North American Association for Computational Linguistics (NAACL)*, pages 320–327.

M. A. Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Interational Conference on Computational linguistics*, pages 539–545.

P. Liang. 2005. Semi-supervised learning for natural language. Master's thesis, Massachusetts Institute of Technology.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space. Technical Report 1301.3781, arXiv.

T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NIPS)*.

J. Pennington, R. Socher, and C. D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

L. Ratinov and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Computational Natural Language Learning (CoNLL)*.

E. Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Association for the Advancement of Artificial Intelligence (AAAI)*, pages 1044–1049.

A. Subramanya, S. Petrov, and F. Pereira. 2010. Efficient graph-based semi-supervised learning of structured tagging models. In *Empirical Methods in Natural Language Processing (EMNLP)*.

M. Thelen and E. Riloff. 2002. A bootstrapping method for learning semantic lexicons using extraction pattern contexts. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 214–221.

M. Whitney and A. Sarkar. 2012. Bootstrapping via graph propagation. In *Association for Computational Linguistics (ACL)*.

R. Yangarber, R. Grishman, and P. Tapanainen. 2000. Automatic acquisition of domain knowledge for information extraction. In *International Conference on Computational Linguistics (COLING)*, pages 940–946.

D. Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Association for Computational Linguistics (ACL)*.