Relation Extraction Using Label Propagation Based Semi-supervised Learning

Jinxiu Chen[1] **Donghong Ji**[1] **Chew Lim Tan**[2] **Zhengyu Niu**[1]

[1]Institute for Infocomm Research
21 Heng Mui Keng Terrace
119613 Singapore
{jinxiu,dhji,zniu}@i2r.a-star.edu.sg

[2]Department of Computer Science
National University of Singapore
117543 Singapore
tancl@comp.nus.edu.sg

Abstract

Shortage of manually labeled data is an obstacle to supervised relation extraction methods. In this paper we investigate a graph based semi-supervised learning algorithm, a label propagation (LP) algorithm, for relation extraction. It represents labeled and unlabeled examples and their distances as the nodes and the weights of edges of a graph, and tries to obtain a labeling function to satisfy two constraints: 1) it should be fixed on the labeled nodes, 2) it should be smooth on the whole graph. Experiment results on the ACE corpus showed that this LP algorithm achieves better performance than SVM when only very few labeled examples are available, and it also performs better than bootstrapping for the relation extraction task.

1 Introduction

Relation extraction is the task of detecting and classifying relationships between two entities from text. Many machine learning methods have been proposed to address this problem, e.g., supervised learning algorithms (Miller et al., 2000; Zelenko et al., 2002; Culotta and Soresen, 2004; Kambhatla, 2004; Zhou et al., 2005), semi-supervised learning algorithms (Brin, 1998; Agichtein and Gravano, 2000; Zhang, 2004), and unsupervised learning algorithms (Hasegawa et al., 2004).

Supervised methods for relation extraction perform well on the ACE Data, but they require a large amount of manually labeled relation instances. Unsupervised methods do not need the definition of relation types and manually labeled data, but they cannot detect relations between entity pairs and its result cannot be directly used in many NLP tasks since there is no relation type label attached to each instance in clustering result. Considering both the availability of a large amount of untagged corpora and direct usage of extracted relations, semi-supervised learning methods has received great attention.

DIPRE (Dual Iterative Pattern Relation Expansion) (Brin, 1998) is a bootstrapping-based system that used a pattern matching system as classifier to exploit the duality between sets of patterns and relations. Snowball (Agichtein and Gravano, 2000) is another system that used bootstrapping techniques for extracting relations from unstructured text. Snowball shares much in common with DIPRE, including the employment of the bootstrapping framework as well as the use of pattern matching to extract new candidate relations. The third system approaches relation classification problem with bootstrapping on top of SVM, proposed by Zhang (2004). This system focuses on the ACE subproblem, RDC, and extracts various lexical and syntactic features for the classification task. However, Zhang (2004)'s method doesn't actually "detect" relaitons but only performs relation classification between two entities given that they are known to be related.

Bootstrapping works by iteratively classifying unlabeled examples and adding confidently classified examples into labeled data using a model learned from augmented labeled data in previous iteration. It can be found that the affinity information among unlabeled examples is not fully explored in this bootstrapping process.

Recently a promising family of semi-supervised learning algorithm is introduced, which can effectively combine unlabeled data with labeled data in learning process by exploiting manifold structure (cluster structure) in data (Belkin and Niyogi, 2002; Blum and Chawla, 2001; Blum et al., 2004; Zhu and Ghahramani, 2002; Zhu et al., 2003). These graph-based semi-supervised methods usually define a graph where the nodes represent labeled and unlabeled examples in a dataset, and edges (may be weighted) reflect the similarity of examples. Then one wants a labeling function to satisfy two constraints at the same time: 1) it should be close to the given labels on the labeled nodes, and 2) it should be smooth on the whole graph. This can be expressed in a regularization framework where the first term is a loss function, and the second term is a regularizer. These methods differ from traditional semi-supervised learning methods in that they use graph structure to smooth the labeling function.

To the best of our knowledge, no work has been done on using graph based semi-supervised learning algorithms for relation extraction. Here we investigate a label propagation algorithm (LP) (Zhu and Ghahramani, 2002) for relation extraction task. This algorithm works by representing labeled and unlabeled examples as vertices in a connected graph, then propagating the label information from any vertex to nearby vertices through weighted edges iteratively, finally inferring the labels of unlabeled examples after the propagation process converges. In this paper we focus on the ACE RDC task[1].

The rest of this paper is organized as follows. Section 2 presents related work. Section 3 formulates relation extraction problem in the context of semi-supervised learning and describes our proposed approach. Then we provide experimental results of our proposed method and compare with a popular supervised learning algorithm (SVM) and bootstrapping algorithm in Section 4. Finally we conclude our work in section 5.

[1] http://www.ldc.upenn.edu/Projects/ACE/, Three tasks of ACE program: Entity Detection and Tracking (EDT), Relation Detection and Characterization (RDC), and Event Detection and Characterization (EDC)

2 The Proposed Method

2.1 Problem Definition

The problem of relation extraction is to assign an appropriate relation type to an occurrence of two entity pairs in a given context. It can be represented as follows:

$$R \rightarrow (C_{pre}, e_1, C_{mid}, e_2, C_{post}) \qquad (1)$$

where e_1 and e_2 denote the entity mentions, and C_{pre},C_{mid},and C_{post} are the contexts before, between and after the entity mention pairs. In this paper, we set the mid-context window as the words between the two entity mentions and the pre- and post-context as up to two words before and after the corresponding entity mention.

Let $X = \{x_i\}_{i=1}^{n}$ be a set of contexts of occurrences of all the entity mention pairs, where x_i represents the contexts of the i-th occurrence, and n is the total number of occurrences. The first l examples (or contexts) are labeled as y_g ($y_g \in \{r_j\}_{j=1}^{R}$, r_j denotes relation type and R is the total number of relation types). The remaining $u(u = n - l)$ examples are unlabeled.

Intuitively, if two occurrences of entity mention pairs have the similarity context, they tend to hold the same relation type. Based on the assumption, we define a graph where the vertices represent the contexts of labeled and unlabeled occurrences of entity mention pairs, and the edge between any two vertices x_i and x_j is weighted so that the closer the vertices in some distance measure, the larger the weight associated with this edge. Hence, the weights are defined as follows:

$$W_{ij} = exp(-\frac{s_{ij}^2}{\alpha^2}) \qquad (2)$$

where s_{ij} is the similarity between x_i and x_j calculated by some similarity measures, e.g., cosine similarity, and α is used to scale the weights. In this paper, we set α as the average similarity between labeled examples from different classes.

2.2 A Label Propagation Algorithm

In the LP algorithm, the label information of any vertex in a graph is propagated to nearby vertices through weighted edges until a global stable stage is achieved. Larger edge weights allow labels to travel through easier. Thus the closer the examples are, the more likely they have similar labels.

We define soft label as a vector that is a probabilistic distribution over all the classes. In the label propagation process, the soft label of each initial labeled example is clamped in each iteration to replenish label sources from these labeled data. Thus the labeled data act like sources to push out labels through unlabeled data. With this push from labeled examples, the class boundaries will be pushed through edges with large weights and settle in gaps along edges with small weights. Hopefully, the values of W_{ij} across different classes would be as small as possible and the values of W_{ij} within the same class would be as large as possible. This will make label propagation to stay within the same class. This label propagation process will make the labeling function smooth on the graph.

Define an $n \times n$ probabilistic transition matrix T

$$T_{ij} = P(j \rightarrow i) = \frac{w_{ij}}{\sum_{k=1}^{n} w_{kj}} \quad (3)$$

where T_{ij} is the probability to jump from vertex x_j to vertex x_i. We define a $n \times R$ label matrix Y, where Y_{ij} representing the probabilities of vertex y_i to have the label r_j.

Then the label propagation algorithm consists the following main steps:

Step1 : Initialization

- Set the iteration index $t = 0$;
- Let Y^0 be the initial soft labels attached to each vertex, where $Y^0_{ij} = 1$ if y_i is label r_j and 0 otherwise.
- Let Y^0_L be the top l rows of Y^0 and Y^0_U be the remaining u rows. Y^0_L is consistent with the labeling in labeled data and the initialization of Y^0_U can be arbitrary.

Step 2 : Propagate the labels of any vertex to nearby vertices by $Y^{t+1} = \overline{T} Y^t$, where $\overline{T}$ is the row-normalized matrix of T, i.e. $\overline{T_{ij}} = T_{ij} / \sum_k T_{ik}$, which can maintain the class probability interpretation.

Step 3 : Clamp the labeled data, that is, replace the top l row of Y^{t+1} with Y^0_L.

Step 4 : Repeat from step 2 until Y converges.

Step 5 : Assign $x_h (l + 1 \leq h \leq n)$ with a label: $y_h = argmax_j Y_{hj}$.

The above algorithm can ensure that the labeled data Y_L never changes since it is clamped in Step 3. Actually we are interested in only Y_U. This algorithm has been shown to converge to a unique solution $\hat{Y}_U = \lim_{t \to \infty} Y^t_U = (I - \bar{T}_{uu})^{-1} \bar{T}_{ul} Y^0_L$ (Zhu and Ghahramani, 2002). Here, $\bar{T}_{uu}$ and $\bar{T}_{ul}$ are acquired by splitting matrix $\bar{T}$ after the l-th row and the l-th column into 4 sub-matrices. And I is $u \times u$ identity matrix. We can see that the initialization of Y^0_U in this solution is not important, since Y^0_U does not affect the estimation of $\hat{Y}_U$.

3 Experiments and Results

3.1 Feature Set

Following (Zhang, 2004), we used lexical and syntactic features in the contexts of entity pairs, which are extracted and computed from the parse trees derived from Charniak Parser (Charniak, 1999) and the Chunklink script [2] written by Sabine Buchholz from Tilburg University.

Words: Surface tokens of the two entities and words in the three contexts.

Entity Type: the entity type of both entity mentions, which can be PERSON, ORGANIZATION, FACILITY, LOCATION and GPE.

POS features: Part-Of-Speech tags corresponding to all tokens in the two entities and words in the three contexts.

Chunking features: This category of features are extracted from the chunklink representation, which includes:

- **Chunk tag information** of the two entities and words in the three contexts. The “0” tag means that the word is not in any chunk. The “I-XP” tag means that this word is inside an XP chunk. The “B-XP” by default means that the word is at the beginning of an XP chunk.
- **Grammatical function** of the two entities and words in the three contexts. The

[2]Software available at http://ilk.uvt.nl/∼sabine/chunklink/

last word in each chunk is its head, and the function of the head is the function of the whole chunk. "NP-SBJ" means a NP chunk as the subject of the sentence. The other words in a chunk that are not the head have "NOFUNC" as their function.

- **IOB-chains** of the heads of the two entities. So-called IOB-chain, noting the syntactic categories of all the constituents on the path from the root node to this leaf node of tree.

The position information is also specified in the description of each feature above. For example, word features with position information include:

1) WE1 (WE2): all words in e_1 (e_2)

2) WHE1 (WHE2): head word of e_1 (e_2)

3) WMNULL: no words in C_{mid}

4) WMFL: the only word in C_{mid}

5) WMF, WML, WM2, WM3, ...: first word, last word, second word, third word, ...in C_{mid} when at least two words in C_{mid}

6) WEL1, WEL2, ...: first word, second word, ... before e_1

7) WER1, WER2, ...: first word, second word, ... after e_2

We combine the above lexical and syntactic features with their position information in the contexts to form context vectors. Before that, we filter out low frequency features which appeared only once in the dataset.

3.2 Similarity Measures

The similarity s_{ij} between two occurrences of entity pairs is important to the performance of the LP algorithm. In this paper, we investigated two similarity measures, cosine similarity measure and Jensen-Shannon (JS) divergence (Lin, 1991). Cosine similarity is commonly used semantic distance, which measures the angle between two feature vectors. JS divergence has ever been used as distance measure for document clustering, which outperforms cosine similarity based document clustering (Slonim et al., 2002). JS divergence measures the distance between two probability distributions if feature vector is considered as probability distribution over features. JS divergence is defined as follows:

Table 1: Frequency of Relation SubTypes in the ACE training and devtest corpus.

Type	SubType	Training	Devtest
ROLE	General-Staff	550	149
	Management	677	122
	Citizen-Of	127	24
	Founder	11	5
	Owner	146	15
	Affiliate-Partner	111	15
	Member	460	145
	Client	67	13
	Other	15	7
PART	Part-Of	490	103
	Subsidiary	85	19
	Other	2	1
AT	Located	975	192
	Based-In	187	64
	Residence	154	54
SOC	Other-Professional	195	25
	Other-Personal	60	10
	Parent	68	24
	Spouse	21	4
	Associate	49	7
	Other-Relative	23	10
	Sibling	7	4
	GrandParent	6	1
NEAR	Relative-Location	88	32

$$JS(q,r) = \frac{1}{2}[D_{KL}(q\|\bar{p}) + D_{KL}(r\|\bar{p})] \tag{4}$$

$$D_{KL}(q\|\bar{p}) = \sum_y q(y)(\log\frac{q(y)}{\bar{p}(y)}) \tag{5}$$

$$D_{KL}(r\|\bar{p}) = \sum_y r(y)(\log\frac{r(y)}{\bar{p}(y)}) \tag{6}$$

where $\bar{p} = \frac{1}{2}(q + r)$ and $JS(q,r)$ represents JS divergence between probability distribution q(y) and r(y) (y is a random variable), which is defined in terms of KL-divergence.

3.3 Experimental Evaluation

3.3.1 Experiment Setup

We evaluated this label propagation based relation extraction method for relation subtype detection and characterization task on the official ACE 2003 corpus. It contains 519 files from sources including broadcast, newswire, and newspaper. We dealt with only intra-sentence explicit relations and assumed that all entities have been detected beforehand in the EDT sub-task of ACE. Table 1 lists the types and subtypes of relations for the ACE Relation Detection and Characterization (RDC) task, along with their

Table 2: The Performance of SVM and LP algorithm with different sizes of labeled data for relation detection on relation subtypes. The LP algorithm is run with two similarity measures: cosine similarity and JS divergence.

	SVM			LP_{Cosine}			LP_{JS}		
Percentage	P	R	F	P	R	F	P	R	F
1%	35.9	32.6	34.4	58.3	56.1	57.1	58.5	58.7	58.5
10%	51.3	41.5	45.9	64.5	57.5	60.7	64.6	62.0	63.2
25%	67.1	52.9	59.1	68.7	59.0	63.4	68.9	63.7	66.1
50%	74.0	57.8	64.9	69.9	61.8	65.6	70.1	64.1	66.9
75%	77.6	59.4	67.2	71.8	63.4	67.3	72.4	64.8	68.3
100%	79.8	62.9	70.3	73.9	66.9	70.2	74.2	68.2	71.1

Table 3: The performance of SVM and LP algorithm with different sizes of labeled data for relation detection and classification on relation subtypes. The LP algorithm is run with two similarity measures: cosine similarity and JS divergence.

	SVM			LP_{Cosine}			LP_{JS}		
Percentage	P	R	F	P	R	F	P	R	F
1%	31.6	26.1	28.6	39.6	37.5	38.5	40.1	38.0	39.0
10%	39.1	32.7	35.6	45.9	39.6	42.5	46.2	41.6	43.7
25%	49.8	35.0	41.1	51.0	44.5	47.3	52.3	46.0	48.9
50%	52.5	41.3	46.2	54.1	48.6	51.2	54.9	50.8	52.7
75%	58.7	46.7	52.0	56.0	52.0	53.9	56.1	52.6	54.3
100%	60.8	48.9	54.2	56.2	52.3	54.1	56.3	52.9	54.6

frequency of occurrence in the ACE training set and test set. We constructed labeled data by randomly sampling some examples from ACE training data and additionally sampling examples with the same size from the pool of unrelated entity pairs for the "NONE" class. We used the remaining examples in the ACE training set and the whole ACE test set as unlabeled data. The testing set was used for final evaluation.

3.3.2 LP vs. SVM

Support Vector Machine (SVM) is a state of the art technique for relation extraction task. In this experiment, we use LIBSVM tool [3] with linear kernel function.

For comparison between SVM and LP, we ran SVM and LP with different sizes of labeled data and evaluate their performance on unlabeled data using precision, recall and F-measure. Firstly, we ran SVM or LP algorithm to detect possible relations from unlabeled data. If an entity mention pair is classified not to the "NONE" class but to the other 24 subtype classes, then it has a relation. Then construct labeled datasets with different sampling set size l, including $1\% \times N_{train}$, $10\% \times N_{train}$, $25\% \times N_{train}$, $50\% \times N_{train}$, $75\% \times N_{train}$, $100\% \times N_{train}$ (N_{train} is the number of examples in the ACE training set). If any relation subtype was absent from the sampled labeled set, we redid the sampling. For each size, we performed 20 trials and calculated average scores on test set over these 20 random trials.

Table 2 reports the performance of SVM and LP with different sizes of labled data for relation detection task. We used the same sampled labeled data in LP as the training data for SVM model.

From Table 2, we see that both LP_{Cosine} and LP_{JS} achieve higher *Recall* than SVM. Specifically, with small labeled dataset (percentage of labeled data $\leq 25\%$), the performance improvement by LP is significant. When the percentage of labeled data increases from 50% to 100%, LP_{Cosine} is still comparable to SVM in *F-measure* while LP_{JS} achieves slightly better *F-measure* than SVM. On the other hand, LP_{JS} consistently outperforms LP_{Cosine}.

Table 3 reports the performance of relation classification by using SVM and LP with different sizes of labled data. And the performance describes the average values of *Precision*, *Recall* and *F-measure* over major relation subtypes.

From Table 3, we see that LP_{Cosine} and LP_{JS} outperform SVM by *F-measure* in almost all settings of labeled data, which is due to the increase of *Recall*. With smaller labeled dataset (percentage of labeled data $\leq 50\%$), the gap between LP and SVM is larger. When the percentage of labeled data in-

[3] $LIBSVM$: a library for support vector machines. Software available at http://www.csie.ntu.edu.tw/∼cjlin/libsvm.

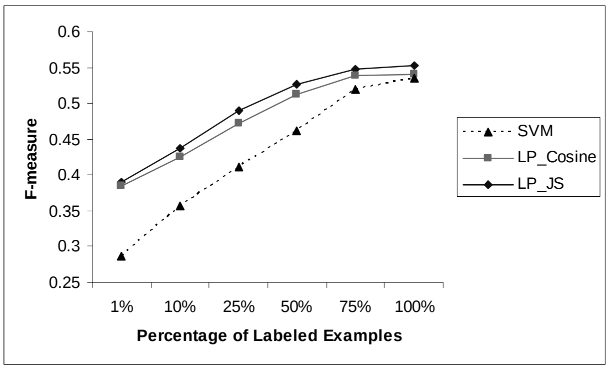

Figure 1: Comparison of the performance of SVM and LP with different sizes of labeled data

creases from 75% to 100%, the performance of LP algorithm is still comparable to SVM. On the other hand, the LP algorithm based on JS divergence consistently outperforms the LP algorithm based on Cosine similarity. Figure 1 visualizes the accuracy of three algorithms.

As shown in Figure 1, the gap between SVM curve and LP$_{JS}$ curves is large when the percentage of labeled data is relatively low.

3.3.3 An Example

In Figure 2, we selected 25 instances in training set and 15 instances in test set from the ACE corpus,which covered five relation types. Using $Isomap$ tool [4], the 40 instances with 229 feature dimensions are visualized in a two-dimensional space as the figure. We randomly sampled only one labeled example for each relation type from the 25 training examples as labeled data. Figure 2(a) and 2(b) show the initial state and ground truth result respectively. Figure 2(c) reports the classification result on test set by SVM ($accuracy = \frac{4}{15} = 26.7\%$), and Figure 2(d) gives the classification result on both training set and test set by LP ($accuracy = \frac{11}{15} = 73.3\%$).

Comparing Figure 2(b) and Figure 2(c), we find that many examples are misclassified from class $\diamond$ to other class symbols. This may be caused that SVMs method ignores the intrinsic structure in data. For Figure 2(d), the labels of unlabeled examples are determined not only by nearby labeled examples, but also by nearby unlabeled examples, so using LP

[4]The tool is available at http://isomap.stanford.edu/.

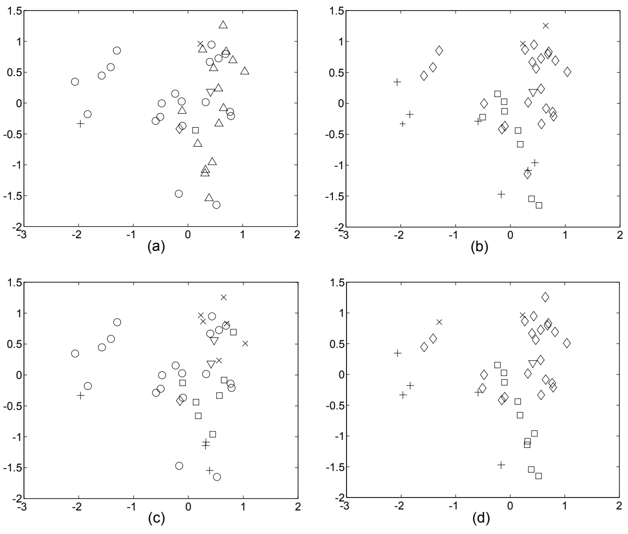

Figure 2: An example: comparison of SVM and LP algorithm on a data set from ACE corpus. $\circ$ and $\triangle$ denote the unlabeled examples in training set and test set respectively, and other symbols ($\diamond, \times, \square, +$ and $\bigtriangledown$) represent the labeled examples with respective relation type sampled from training set.

strategy achieves better performance than the local consistency based SVM strategy when the size of labeled data is quite small.

3.3.4 LP vs. Bootstrapping

In (Zhang, 2004), they perform relation classification on ACE corpus with bootstrapping on top of SVM. To compare with their proposed Bootstrapped SVM algorithm, we use the same feature stream setting and randomly selected 100 instances from the training data as the size of initial labeled data.

Table 4 lists the performance of the bootstrapped SVM method from (Zhang, 2004) and LP method with 100 seed labeled examples for relation type classification task. We can see that LP algorithm outperforms the bootstrapped SVM algorithm on four relation type classification tasks, and perform comparably on the relation ”SOC” classification task.

4 Discussion

In this paper,we have investigated a graph-based semi-supervised learning approach for relation extraction problem. Experimental results showed that the LP algorithm performs better than SVM and

Table 4: Comparison of the performance of the bootstrapped SVM method from (Zhang, 2004) and LP method with 100 seed labeled examples for relation type classification task.

	Bootstrapping			LP_{JS}		
Relation type	P	R	F	P	R	F
ROLE	78.5	69.7	73.8	81.0	74.7	77.7
PART	65.6	34.1	44.9	70.1	41.6	52.2
AT	61.0	84.8	70.9	74.2	79.1	76.6
SOC	47.0	57.4	51.7	45.0	59.1	51.0
NEAR	–	–	–	13.7	12.5	13.0

Table 5: Comparison of the performance of previous methods on ACE RDC task.

		Relation Dectection			Relation Detection and Classification					
					on Types			on Subtypes		
	Method	P	R	F	P	R	F	P	R	F
Culotta and Soresen (2004)	Tree kernel based	81.2	51.8	63.2	67.1	35.0	45.8	-	-	-
Kambhatla (2004)	Feature based, Maximum Entropy	-	-	-	-	-	-	63.5	45.2	52.8
Zhou et al. (2005)	Feature based,SVM	84.8	66.7	74.7	77.2	60.7	68.0	63.1	49.5	55.5

bootstrapping. We have some findings from these results:

The LP based relation extraction method can use the graph structure to smooth the labels of unlabeled examples. Therefore, the labels of unlabeled examples are determined not only by the nearby labeled examples, but also by nearby unlabeled examples. For supervised methods, e.g., SVM, very few labeled examples are not enough to reveal the structure of each class. Therefore they can not perform well, since the classification hyperplane was learned only from few labeled data and the coherent structure in unlabeled data was not explored when inferring class boundary. Hence, our LP-based semi-supervised method achieves better performance on both relation detection and classification when only few labeled data is available. Bootstrapping

Currently most of works on the RDC task of ACE focused on supervised learning methods Culotta and Soresen (2004; Kambhatla (2004; Zhou et al. (2005). Table 5 lists a comparison on relation detection and classification of these methods. Zhou et al. (2005) reported the best result as 63.1%/49.5%/55.5% in *Precision/Recall/F-measure* on the relation subtype classification using feature based method, which outperforms tree kernel based method by Culotta and Soresen (2004). Compared with Zhou et al.'s method, the performance of LP algorithm is slightly lower. It may be due to that we used a much simpler feature set. Our work in this paper focuses on the investigation of a graph based semi-supervised learning algorithm for relation extraction. In the future, we would like to use more effective feature sets Zhou et al. (2005) or kernel based similarity measure with LP for relation extraction.

5 Conclusion and Future Work

This paper approaches the problem of semi-supervised relation extraction using a label propagation algorithm. It represents labeled and unlabeled examples and their distances as the nodes and the weights of edges of a graph, and tries to obtain a labeling function to satisfy two constraints: 1) it should be fixed on the labeled nodes, 2) it should be smooth on the whole graph. In the classification process, the labels of unlabeled examples are determined not only by nearby labeled examples, but also by nearby unlabeled examples. Our experimental results demonstrated that this graph based algorithm can achieve better performance than SVM when only very few labeled examples are available, and also outperforms the bootstrapping method for relation extraction task.

In the future, we would like to investigate more effective feature set or use feature selection to improve the performance of this graph-based semi-supervised relation extraction method.

References

Agichtein E. and Gravano L.. 2000. *Snowball: Extracting Relations from large Plain-Text Collections, In Proceedings of the 5^{th} ACM International Conference on Digital Libraries (ACMDL'00)*.

Belkin M. and Niyogi P.. 2002. *Using Manifold Structure for Partially Labeled Classification. Advances in Neural Infomation Processing Systems 15*.

Blum A. and Chawla S. 2001. *Learning from Labeled and Unlabeled Data Using Graph Mincuts. In Proceedings of the 18th International Conference on Machine Learning*.

Blum A., Lafferty J., Rwebangira R. and Reddy R. 2004. *Semi-Supervised Learning Using Randomized Mincuts. In Proceedings of the 21th International Conference on Machine Learning.*.

Brin Sergey. 1998. *Extracting patterns and relations from world wide web. In Proceedings of WebDB Workshop at 6th International Conference on Extending Database Technology (WebDB'98)*. pages 172-183.

Charniak E. 1999. *A Maximum-entropy-inspired parser. Technical Report CS-99-12*. Computer Science Department, Brown University.

Culotta A. and Soresen J. 2004. *Dependency tree kernels for relation extraction, In Proceedings of 42th Annual Meeting of the Association for Computational Linguistics*. 21-26 July 2004. Barcelona, Spain.

Hasegawa T., Sekine S. and Grishman R. 2004. *Discovering Relations among Named Entities from Large Corpora, In Proceeding of Conference ACL2004*. Barcelona, Spain.

Kambhatla N. 2004. *Combining lexical, syntactic and semantic features with Maximum Entropy Models for extracting relations, In Proceedings of 42th Annual Meeting of the Association for Computational Linguistics.*. 21-26 July 2004. Barcelona, Spain.

Lin J. 1991. *Divergence Measures Based on the Shannon Entropy. IEEE Transactions on Information Theory*. Vol 37, No.1, 145-150.

Miller S.,Fox H.,Ramshaw L. and Weischedel R. 2000. *A novel use of statistical parsing to extract information from text. In Proceedings of 6th Applied Natural Language Processing Conference* 29 April-4 may 2000, Seattle USA.

Slonim, N., Friedman, N., and Tishby, N. 2002. *Unsupervised Document Classification Using Sequential Information Maximization. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Yarowsky D. 1995. *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods. In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*. pp.189-196.

Zelenko D., Aone C. and Richardella A. 2002. *Kernel Methods for Relation Extraction, Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Philadelphia.

Zhang Zhu. 2004. *Weakly-supervised relation classification for Information Extraction, In Proceedings of ACM 13th conference on Information and Knowledge Management (CIKM'2004)*. 8-13 Nov 2004. Washington D.C.,USA.

Zhou GuoDong, Su Jian, Zhang Jie and Zhang min. 2005. *Exploring Various Knowledge in Relation Extraction. In Proceedings of 43th Annual Meeting of the Association for Computational Linguistics*. USA.

Zhu Xiaojin and Ghahramani Zoubin. 2002. *Learning from Labeled and Unlabeled Data with Label Propagation. CMU CALD tech report CMU-CALD-02-107*.

Zhu Xiaojin, Ghahramani Zoubin, and Lafferty J. 2003. *Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In Proceedings of the 20th International Conference on Machine Learning*.