

Chinese Chunking with Tri-Training Learning

Wenliang Chen^{†‡}, Yujie Zhang[†], and Hitoshi Isahara[†]

[†]Computational Linguistics Group
National Institute of Information and Communications Technology
3-5 Hikari-dai, Seika-cho, Soraku-gun, Kyoto, Japan, 619-0289

[‡]Natural Language Processing Lab
Northeastern University, Shenyang, China, 110004
{chenwl, yujie, isahara}@nict.go.jp

Abstract. This paper presents a practical tri-training method for Chinese chunking using a small amount of labeled training data and a much larger pool of unlabeled data. We propose a novel selection method for tri-training learning in which newly labeled sentences are selected by comparing the agreements of three classifiers. In detail, in each iteration, a new sample is selected for a classifier if the other two classifiers agree on the labels while itself disagrees. We compare the proposed tri-training learning approach with co-training learning approach on Upenn Chinese Treebank V4.0(CTB4). The experimental results show that the proposed approach can improve the performance significantly.

1 Introduction

Chunking identifies the non-recursive cores of various types of phrases in text, possibly as a precursor to full parsing or information extraction. Steven P. Abney was the first person to introduce chunks for parsing[1]. Ramshaw and Marcus[2] first represented base noun phrase recognition as a machine learning problem. In 2000, CoNLL-2000 introduced a shared task to tag many kinds of phrases besides noun phrases in English[3]. Much work has been done on Chinese chunking[4–7], of which supervised learning approaches are the most successful. However, to achieve good performance, the amount of manually labeled training data required by supervised learning methods is quite large.

Semi-supervised learning has recently become an active research area. It requires only a small amount of labeled training data and improves performance using unlabeled data. In this paper, we investigate the use of a semi-supervised learning approach, tri-training learning[8], on Chinese chunking. By considering that Chinese chunking is a sequence labeling problem, we propose a novel approach of selecting training samples for tri-training learning. The experimental results show that the proposed approach can improve the performance significantly using a large pool of unlabeled data.

The rest of this paper is as follows: Section 2 describes the definition of Chinese chunking. Section 3 describes the algorithm of tri-training for Chinese chunking. Section 4 introduces the related works. Section 5 explains the experimental results. Finally, in section 6 we draw the conclusions.

2 Chinese Chunking

We defined Chinese chunks based on the Upenn Chinese Treebank V4.0 (CTB4)¹ as Chen et al. [9] did. And we use a tool² to generate the Chinese chunk dataset from CTB4.

2.1 Data Representation

To represent the chunks clearly, we represent the data with an IOB-based model as the CoNLL-2000 shared task did, in which every word is to be tagged with a chunk type label extended with I (inside a chunk), O (outside a chunk), and B (inside a chunk, but also the first word of the chunk).

With data representation, the problem of Chinese chunking can be regarded as a sequence labeling task. That is to say, given a sequence of tokens (words pairing with Part-of-Speech tags), $x = \{x_1, x_2, \dots, x_n\}$, we need to generate a sequence of chunk tags, $y = \{y_1, y_2, \dots, y_n\}$. In the following sections, we call an original sentence (x) as a data sequence, and a labeled sentence (y) as a labeled sequence.

2.2 Features

In this paper, we regard Chinese chunking as a sequence labeling problem. The observations are based on features that are able to represent the difference between two events. We utilize both lexical and Part-Of-Speech (POS) information as the features.

We use the lexical and POS information within a fixed window. The features are listed as follows:

- $W_i (i = -2, -1, 0, 1, 2)$
- $W_i W_{i+1} (i = -2, -1, 0, 1)$
- $P_i (i = -2, -1, 0, 1, 2)$
- $P_i P_{i+1} (i = -2, -1, 0, 1)$

Where W refers to a Chinese word while W_0 denotes the current word and $W_i (W_{-i})$ denotes the word i positions to the right (left) of the current word, and P refers to a POS tag while P_0 denotes the current POS and $P_i (P_{-i})$ denotes the POS i positions to the right (left) of the current POS.

3 Tri-training for Chinese Chunking

Tri-training was proposed by Zhou and Li [8], which was motivated from co-training. It designs three classifiers learning from unlabeled examples via an unlabeled example is labeled for a classifier if the other two classifiers agree on the labeling under certain conditions. This method can release the requirement of co-training with sufficient and redundant views. Additionally, tri-training learning considers the agreements of the classifiers while selecting new samples. We just need the labeled tags instead of the confident scores made by the classifiers. Thus we can use any classifier, which can be used in chunking, in tri-training learning.

¹ More detailed information at <http://www.cis.upenn.edu/chinese/>.

² Tool is available at <http://www.nlplab.cn/chenwl/tools/chunklinkctb.txt>

3.1 Algorithm of Tri-training

The algorithm of tri-training for chunking is presented in Table 1, and consists of three different classifiers. At each iteration, the unlabeled sentences are labeled by the current classifiers. Next, a subset of the sentences newly labeled is selected to be added to the training data. The general control flow of the algorithm is similar to the algorithm described by [8]. However, there are some differences in our algorithm. Firstly, we design a new measure to compute the agreement between two classifiers. Secondly, we propose a novel selection method based on the agreement measure.

Table 1. The pseudo-code for the Tri-training Algorithm

A, B, and C are three different classifiers.
 M_A^i , M_B^i and M_C^i are the models for A, B and C at step i.
 L is the original labeled data and U is the original unlabeled data.
 U_A^i , U_B^i , and U_C^i are the unlabeled data at step i.
 L_A^i , L_B^i , and L_C^i are the labeled training data for A, B and C at step i.

Initialise:
 $L_A^0 = L_B^0 = L_C^0 = L$.
 $U_A^0 = U_B^0 = U_C^0 = U$.
 $M_A^0 \leftarrow Train(A, L_A^0)$.
 $M_B^0 \leftarrow Train(B, L_B^0)$.
 $M_C^0 \leftarrow Train(C, L_C^0)$.

Loop: {
 M_A^i , M_B^i and M_C^i tag the sentences in U_A^i , U_B^i , and U_C^i .
 Select newly labeled sentences L_A^{new} , L_B^{new} and L_C^{new} according to some selection method S .
 $L_A^{i+1} = L_A^i + L_A^{new}$, $U_A^{i+1} = U_A^i - L_A^{new}$.
 $L_B^{i+1} = L_B^i + L_B^{new}$, $U_B^{i+1} = U_B^i - L_B^{new}$.
 $L_C^{i+1} = L_C^i + L_C^{new}$, $U_C^{i+1} = U_C^i - L_C^{new}$.
 $M_A^{i+1} \leftarrow Train(A, L_A^{i+1})$.
 $M_B^{i+1} \leftarrow Train(B, L_B^{i+1})$.
 $M_C^{i+1} \leftarrow Train(C, L_C^{i+1})$.
 if (all L_A^{new} , L_B^{new} and L_C^{new} are \emptyset) End Loop.
}

3.2 Select Training Samples

In selecting new training samples procedure, there are two steps: 1) compute the scores for all sentences; 2) select some sentences as new training samples according to the scores. We design a simple agreement measure to compute the scores for all sentences. Then based on the agreement measure, we propose a novel sample selection method, which is called Two Agree One Disagree method, to select newly labeled sentence as training samples.

Agreement Measure Here, we describe how to compute the agreement between two classifiers for every sentence.

Suppose we have a data sequence $x = \{x_1, \dots, x_n\}$. Then use two classifiers to tag the sequence x , to get two labeled sequences $y_a = \{y_{1a}, \dots, y_{na}\}$ and $y_b = \{y_{1b}, \dots, y_{nb}\}$. Now, we compute the agreement A_g between y_a and y_b as follows:

$$A_g(y_a, y_b) = \frac{\sum_{1 \leq i \leq n} f(y_{ia}, y_{ib})}{n}. \quad (1)$$

where n is the number of tokens in x , f is a binary function to tell whether y_{ia} and y_{ib} are the same label.

$$f(y_{ia}, y_{ib}) = \begin{cases} 1 & : y_{ia} = y_{ib} \\ 0 & : y_{ia} \neq y_{ib}. \end{cases} \quad (2)$$

A_g denotes the agreement between two labeled sequences or the agreement between two classifiers. The larger A_g is, the higher the agreement is.

Selection Methods After computing the agreement, we should select new samples from the set of newly labeled sentences for next iteration. Suppose have three classifiers A, B, and C, we will select new samples for classifier A. As [10] suggested, we should select the sentences, which have high training utility. Thus we prefer to choose a sentence, which is correctly labeled by B or C and is not parsed correctly by the target classifier A, to be a new training sample.

We adopt two selecting principles: 1) If the higher agreement scores between the classifiers B and C at a sentence, the sentence is more likely correctly labeled. 2) If the classifier A disagree with the other classifiers (B and C) at a sentence, the sentence is not parsed correctly by A. To investigate how the selection criteria affect the learning process, we consider two selection methods.

We propose Two Agree method by applying the first principle. A newly labeled sentence is selected by:

- Two Agree Method (S_{2A}): Firstly, we tag U_A using B and C. Then we compute the agreements of all sentences by Equation 1. Finally, we rank the sentences by the agreements and select top m as new samples for A. The new samples are labeled by B.

By applying both two principles, we propose Two Agree One Disagree Method. A newly labeled sentence is selected by:

- Two Agree One Disagree Method (S_{2A1D}): Firstly, we tag U_A using A, B and C. Then we compute the agreements of A and B for all sentences, also compute the agreements of B and C for all sentences. Then we apply the intersection: the set of m percent highest-agreement scoring labeled sentences by B and C, and the set of m percent lowest-agreement scoring labeled sentences by A and B. The new samples are labeled by B.

Each selection method has a control parameter m , that determines the number of newly labeled sentences to add at each iteration. It also serves as an indirect control of the number of errors added to the training set[10]. In our experiments, we set m as 30% after tuning parameters.

4 Related Works

Semi-supervised learning is halfway between supervised and unsupervised learning. In addition to labeled data, the algorithm requires unlabeled data. The interest in semi-supervised learning increased recently for natural language processing[11–13].

There are some researchers who applied semi-supervised learning on chunking and parsing. Ando and Zhang[14] proposed a semi-supervised learning method that employs the structural learning for English chunking. Steedman et al.[15] learned the statistical parsers from small datasets using bootstrapping.

A prominent semi-supervised learning algorithm is co-training, which was first introduced in Blum and Mitchell[16] as a bootstrapping method. It has the similar control flow as tri-training. In this paper, we also investigate the application of co-training to Chinese chunking. The co-training algorithm used was presented in [11]. We do not use the confident scores. Instead, we select new samples, which have higher agreement score A_g of two classifiers.

5 Experiments

5.1 Experimental Setting

The UPENN Chinese Treebank-4(CTB4) consists of 838 files. In the experiments, we used the first 728 files (FID from chtb_001.fid to chtb_899.fid) as training data, and the other 110 files as testing data. In the training data, we used sizes of labeled sentences: 500 sentences. The other sentences (9,378) were used as unlabeled data for semi-supervised learning methods.

In the experiments, we used Support Vector Machines (SVMs), Conditional Random Fields (CRFs), and Memory-based Learning (MBL)[17] as the classifiers in tri-training. The first two models have achieved good performance in chunking[18][19]. Although the MBL model does not perform well as the first two models do, we hope that it can help to improve the performance of the first two models. And we used SVMs and CRFs as the classifiers in co-training. We used YamCha (V0.33)³ to implement the SVMs model, CRF++ (V0.42)⁴ to the CRFs model, and TiMBL⁵ to the MBL model.

And we used all the default parameter settings of the package in our experiments. In learning procedures, we trained the models in 50 iteration rounds and selected 50 (maximum) newly labeled sentences as new training samples in each iteration. We evaluated the results as CONLL-2000 share-task did. In this paper, we report the results with F_1 score.

5.2 Experimental 1: Selection Methods of Tri-training

In this experiment, we investigated the performance of different selection methods: Two Agree method (S_{2A}) and Two Agree One Disagree method (S_{2A1D}). Figure 1 shows the experimental results, where CRF_S1 refers to the CRFs model with Two Agree method, CRF_S2 refers to the CRFs model with Two Agree One Disagree method, and the other strings have similar meanings.

From the figure, we found that Two Agree One Disagree method achieved better performance than Two Agree method. All three classifiers with Two Agree One Disagree method provided better results. And the CRFs model provided the best results

³ YamCha is available at <http://chasen.org/taku/software/yamcha/>

⁴ CRF++ is available at <http://chasen.org/taku/software/CRF++/>

⁵ TiMBL is available at <http://ilk.uvt.nl/timbl/>

among three models. These results indicated that Two Agree One Disagree method can select new samples more efficiently via applying the second selecting principle. We also found that the MBL model can help to improve the performance of the CRFs and SVMs models, although it did not perform well as the CRFs and SVMs models did.

With Two Agree One Disagree method, tri-training boosts the performance of the MBL model from 83.32% to 86.43% (3.11% higher), the SVMs model from 85.92% to 87.06% (1.11% higher), and the CRFs model from 86.68% to 87.57% (0.89% higher).

5.3 Experimental 2: Tri-training vs Co-training

In this experiment, we compared the performance between tri-training and co-training. Figure 2 shows the comparative results, where tri-training used Two Agree One Disagree method. We found that tri-training outperformed co-training. CRFs with tri-training achieved the best results with 87.57%, while CRFs with co-training got 87.16%.

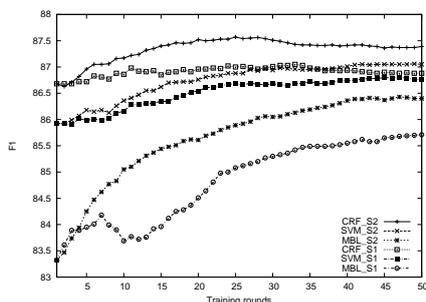


Fig. 1. Results of different selection methods for Tri-training

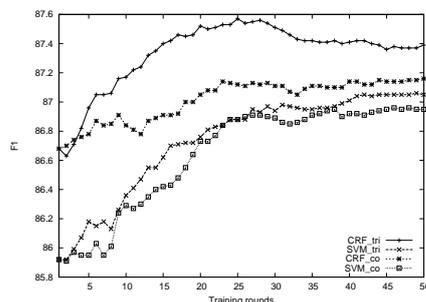


Fig. 2. Tri-training vs Co-training

5.4 discussion

Our experimental results showed that tri-training learning approaches can exploit unlabeled data to improve the performance of Chinese chunking.

To test whether the improvements obtained by tri-training are significant, we performed one-tail paired t-test to compare the accuracy of tri-training against co-training and baseline systems. Here baseline systems refer to the models trained with seed data. For each token, if a classifier gives the correct tag, its score is 1, otherwise its score is 0. The p-values of one-tail paired t-test are shown in Table 2. From the table, we found that tri-training gave higher accuracy than the baseline at the level of significance 0.001 for both the CRFs and SVMs models. For the CRFs model, tri-training provided higher accuracy than co-training at the level of significance 0.01. In addition, for the CRFs model, co-training gave higher accuracy than the baseline at the level of significance 0.01.

Tri-training showed statistically significant improvement over the baseline, but the gap between them was not so big. We looked into the detail over all types of chunks.

Table 2. Summary of the p-value of one-tail paired t-test

nul hypothesis	t-test p-value
baseline vs. Co-training (CRFs)	0.0014
baseline vs. Tri-training (CRFs)	< 0.001
Co-training vs Tri-training(CRFs)	0.0012
baseline vs. Co-training (SVMs)	< 0.001
baseline vs. Tri-training (SVMs)	< 0.001
Co-training vs Tri-training(SVMs)	0.4623

Table 3 shows the results of all types of chunks generated by the CRFs model. Please note that the results of CLP and LST are 0.00 because there are not these two types of tags in 500 original labeled sentences. The results showed that the improvement was not observed uniformly on all types of chunks. Comparing tri-training with baseline systems, there were five types unchanged accuracy or changed very small, one type degraded, and six types changed from 0.2% to 3.35%. These indicated that we should try to improve the performance over all types to get better results.

Table 3. Results on all types of chunks (CRFs)

	baseline	Co-training	Tri-training
ADJP	78.70	79.26	80.09
ADVP	75.87	77.96	76.03
CLP	0.00	0.00	0.00
DNP	99.65	99.66	99.65
DP	99.67	99.70	99.70
DVP	84.16	60.11	86.98
LCP	99.85	99.80	99.82
LST	0.00	0.00	0.00
NP	86.16	86.51	87.36
PP	99.67	99.67	99.67
QP	94.66	94.77	95.89
VP	81.73	82.55	82.40
All	86.68	87.16	87.57

6 Conclusions

This paper presented an experimental study in which three classifiers were trained on labeled and unlabeled data using tri-training. We proposed a novel approach of selecting training samples for tri-training learning by considering the agreements of three classifiers. The experimental results showed that the proposed approach can improve the performance significantly.

In this paper, we trained the models based on word-based sentences with POS tags attached. However in real applications, the input is character-based sentences. Thus in future we will investigate the performance of tri-training learning method on character-based chunking.

References

1. Abney, S.P.: Parsing by chunks. In Berwick, R.C., Abney, S.P., Tenny, C., eds.: *Principle-Based Parsing: Computation and Psycholinguistics*. Kluwer, Dordrecht (1991) 257–278
2. Ramshaw, L., Marcus, M.: Text chunking using transformation-based learning. In Yarovsky, D., Church, K., eds.: *Proceedings of the Third Workshop on Very Large Corpora*, Somerset, New Jersey, Association for Computational Linguistics (1995) 82–94
3. Sang, E.F.T.K., Buchholz, S.: Introduction to the conll-2000 shared task: Chunking. In: *Proceedings of CoNLL-2000 and LLL2000*, Lisbon, Portugal (2000) 127–132
4. Li, H., Webster, J.J., Kit, C., Yao, T.: Transductive hmm based chinese text chunking. In: *Proceedings of IEEE NLP-KE2003*, Beijing, China (2003) 257–262
5. Tan, Y., Yao, T., Chen, Q., Zhu, J.: Applying conditional random fields to chinese shallow parsing. In: *Proceedings of CICLing-2005*, Mexico City, Mexico, Springer (2005) 167–176
6. Wu, S.H., Shih, C.W., Wu, C.W., Tsai, T.H., Hsu, W.L.: Applying maximum entropy to robust chinese shallow parsing. In: *Proceedings of ROCLING2005*. (2005)
7. Zhao, T., Yang, M., Liu, F., Yao, J., Yu, H.: Statistics based hybrid approach to chinese base phrase identification. In: *Proceedings of Second Chinese Language Processing Workshop*. (2000)
8. Zhou, Z.H., Li, M.: Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on Knowledge and Data Engineering* **17** (2005) 1529–1541
9. Chen, W., Zhang, Y., Isahara, H.: An empirical study of chinese chunking. In: *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*, Sydney, Australia, Association for Computational Linguistics (2006) 97–104
10. Steedman, M., Hwa, R., Clark, S., Osborne, M., Sarkar, A., Hockenmaier, J., Ruhlen, P., Baker, S., Crim, J.: Example selection for bootstrapping statistical parsers. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1* (2003) 157–164
11. Pham, T., Ng, H., Lee, W.: Word sense disambiguation with semi-supervised learning. In: *AAAI-05, The Twentieth National Conference on Artificial Intelligence*. (2005)
12. Yarovsky, D.: Unsupervised word sense disambiguation rivaling supervised methods. In: *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL)*. (1995)
13. Collins, M., Singer, Y.: Unsupervised models for named entity classification. *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora* (1999) 100–110
14. Ando, R., Zhang, T.: A high-performance semi-supervised learning method for text chunking. *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)* (2005)
15. Steedman, M., Osborne, M., Sarkar, A., Clark, S., Hwa, R., Hockenmaier, J., Ruhlen, P., Baker, S., Crim, J.: Bootstrapping statistical parsers from small datasets. *The Proceedings of the Annual Meeting of the European Chapter of the ACL* (2003) 331–338
16. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. *Proceedings of the eleventh annual conference on Computational learning theory* (1998) 92–100
17. Sang, E.F.T.K.: Memory-based shallow parsing. *JMLR* **2** (2002) 559–594
18. Sha, F., Pereira, F.: Shallow parsing with conditional random fields. In: *Proceedings of HLT-NAACL03*. (2003)
19. Kudo, T., Matsumoto, Y.: Chunking with support vector machines. In: *Proceedings of NAACL01*. (2001)