Voting between Multiple Data Representations for Text Chunking[⋆]

Hong Shen and Anoop Sarkar

School of Computing Science
Simon Fraser University
Burnaby, BC V5A 1S6, Canada
{hshen,anoop}@cs.sfu.ca

Abstract. This paper considers the hypothesis that voting between *multiple data representations* can be more accurate than voting between *multiple learning models*. This hypothesis has been considered before (cf. [San00]) but the focus was on voting methods rather than the data representations. In this paper, we focus on choosing specific data representations combined with simple majority voting. On the community standard CoNLL-2000 data set, using no additional knowledge sources apart from the training data, we achieved 94.01 $F_{\beta=1}$ score for arbitrary phrase identification compared to the previous best $F_{\beta=1}$ 93.90. We also obtained 95.23 $F_{\beta=1}$ score for Base NP identification. Significance tests show that our Base NP identification score is significantly better than the previous comparable best $F_{\beta=1}$ score of 94.22. Our main contribution is that our model is a fast linear time approach and the previous best approach is significantly slower than our system.

1 Introduction

Text chunking or *shallow parsing* is the task of finding non-recursive phrases in a given sentence of natural language text. Due to the fact that the phrases are assumed to be non-overlapping, the phrase boundaries can be treated as labels, one per word in the input sentence, and *sequence learning* or *sequence prediction* techniques such as the source-channel approach over n-grams can be used to find the most likely sequence of such labels. This was the method introduced in [RM95] where noun phrase chunks were encoded as labels (or tags) on words: **I** for words that are inside a noun chunk, **O** for words outside a chunk, and **B** for a word on the boundary, i.e a word that immediately follows a word with an **I** tag. Chunking was defined as the prediction of a sequence of **I**, **B**, and **O** labels given a word sequence as input. Additional information such as part of speech tags are often used in this prediction. This paper concentrates on the text chunking task: both Base noun phrase (NP) chunking [RM95] and the CoNLL-2000 arbitrary phrase chunking task [SB00]. The CoNLL-2000 shared task considers other chunk types in addition to noun phrases (NPs), such as verb phrases (VPs), among others. The advantage in using these data sets is that they are freely available for experimental comparisons[1].

[⋆] We would like to thank Fred Popowich and the anonymous reviewers for their comments.

[1] From http://lcg-www.uia.ac.be/~erikt/research/np-chunking.html and http://cnts.uia.ac.be/conll2000/chunking/

As a result, there are close to 30 publications that have reported results on these two data sets indicating that improvements on these tasks or even matching the best result is very unlikely to occur using obvious or simple methods.

In order to focus on the specific contribution of multiple data representations, we use a simple trigram model for tagging and chunking[2]. We show that a trigram model for chunking combined with voting between multiple data representations can obtain results equal to the best on the community standard CoNLL-2000 text chunking data set. Using no additional knowledge sources apart from the training data, we achieved 94.01 $F_{\beta=1}$ score for arbitrary phrase identification compared to the previous best $F_{\beta=1}$ 93.90 [KM01][3]. The highest score reported on the CoNLL-2000 data set is 94.17% by [ZDJ02] but this result used a full-fledged parser as an additional knowledge source. Without the parser, their score was 93.57%. We also obtained 95.23 $F_{\beta=1}$ score for Base NP identification. Significance tests show that our Base NP identification score is significantly better than the previous comparable state-of-the-art $F_{\beta=1}$ of 94.22 [KM01][4].

While voting is a commonly used method, it is commonly used as a means to combine the output of multiple machine learning systems. Like [San00] and [KM01] we examine voting between *multiple data representations*. However, we focus purely on the advantage of carefully chosen data representations. We use simple majority voting and a trigram chunking model to focus on the issue of the choice of data representation. Our results show that our approach outperforms other most other voting approaches and is comparable to the previous best approach on the same dataset [KM01]. In addition, as we will show when we compare our approach to [KM01] our main contribution is that our model is a fast linear time approach. [KM01] uses multiple Support Vector Machine classifiers which is slower by a significant order of magnitude.

Based on our empirical results, we show that choosing the right representation (or the types of features used) can be a very powerful alternative in sequence prediction, even when used with relatively simple machine learning methods.

2 The task: Text Chunking

The text chunking problem partitions input sentences into syntactically related non-overlapping groups or chunks. For example, the sentence:

In early trading in Hong Kong Monday , gold was quoted at $ 366.50 an ounce .

can be segmented into the following noun phrase chunks:

[2] Explained in many textbooks. See [Cha96], p.43-56.

[3] The score commonly quoted from [KM01] is 93.91% but this score was based on the IOE1 representation *on the test data* (see explanation in Section 3). In any case, their method provides almost identical accuracy across all representations, but comparable results can be shown only in the IOB2 representation.

[4] Some other approaches obtained their Base NP scores by first performing the arbitrary phrase chunking task and throwing away all other phrases except noun phrases. We do not compare against those scores here for simplicity, since it is technically a different task. Although we are competitive in this other comparison as well.

In (early trading) in (Hong Kong) (Monday) , (gold) was quoted at ($ 366.50) (an ounce) .

The CoNLL-2000 shared task [SB00] was set up as a more general form of the text chunking problem, with additional types in addition to noun phrase chunk types. The CoNLL-2000 data has 11 different chunk types: $\mathcal{T} = \{$ ADJP, ADVP, CONJP, INTJ, LST, NP, PP, PRT, SBAR, VP, UCP $\}$. However, it is important to note that despite the large number of chunk types, the NP, VP and PP types account for 95% of all chunk occurrences. The chunk tags in the data are represented the following three types of tags:

B-t first word of a chunk of type $t \in \mathcal{T}$
I-t non-initial word of a chunk of type $t \in \mathcal{T}$
O word outside of any chunk

The CoNLL-2000 data was based on the noun phrase chunk data extracted by [RM95] from the Penn Treebank. It contains WSJ sections 15-18 of the Penn Treebank (211727 tokens) as training data and section 20 of the Treebank (47377 tokens) as test data. The difference between the two is that Base NP chunking results are evaluated in IOB1 format, while CoNLL-2000 shared task results are evaluated in IOB2 format (see Section 3 for definitions of IOB1 and IOB2). The test data set is processed with a part of speech (POS) tagger (for details, see [SB00]) in order to provide additional information. For example, the above sentence would be assigned POS tags from the Penn Treebank POS tagset as follows:

In_IN early_JJ trading_NN in_IN Hong_NNP Kong_NNP Monday_NNP ,_, gold_NN was_VBD quoted_VBN at_IN $_$ 366.50_CD an_DT ounce_IN ._.

Given the high accuracy of POS tagging, only about 3% to 4% of the tokens are expected to be mistagged in the test data. Evaluation for this task is based on three figures: *precision* (P) is the percentage of detected phrases that are correct, *recall* (R) is the percentage of phrases in the data that were detected, and the $F_{\beta=1}$ score which is the harmonic mean of precision and recall.[5] In this paper, these values are computed using the standard evaluation script that is distributed along with the CoNLL-2000 data set.

3 Multiple Data Representations

We use the following different categories of data representations for the text chunking task. In each case, we describe the non-overlapping chunks with one of the following tag representations and then append the chunk type as a suffix, except for the **O** tag. An example that shows all these representations is provided in Figure 1.

[5] $F_\beta = \frac{(\beta^2+1)pr}{\beta^2 p + r}$

word	IOB1	IOB2	IOE1	IOE2	O+C
In	O	O	O	O	O
early	I	B	I	I	B
trading	I	I	I	E	E
in	O	O	O	O	O
Hong	I	B	I	I	B
Kong	I	I	E	E	E
Monday	B	B	I	E	S
,	O	O	O	O	O
gold	I	B	I	E	S
was	O	O	O	O	O
quoted	O	O	O	O	O
at	O	O	O	O	O
$	I	B	I	I	B
366.50	I	I	E	E	E
an	B	B	I	I	B
ounce	I	I	I	E	E
.	O	O	O	O	O

Fig. 1. The noun chunk tag sequences for the example sentence, *In early trading in Hong Kong Monday , gold was quoted at $ 366.50 an ounce* . is shown represented in five different data representations. We only show noun phrase chunks in this example, all tags except **O** tags will include the type suffix for chunks of different types.

3.1 The Inside/Outside Representation

This representation from [RM95] represents chunks with three tags as follows:

I Current token is inside chunk
B Current token is on the boundary, starting a new chunk with previous token tagged with **I**
O Current token is outside any chunk

[SV99] introduced some variants of this IOB representation: the above representation was now called IOB1. The new variants were called IOB2, IOE1 and IOE2. IOB2 differs from IOB1 in the assignment of tag **B** to every chunk-initial token regardless of whether the next token is inside a chunk. IOE1 differs from IOB1 in that instead of tag **B**, a tag **E** is used for the final token of a chunk that is immediately followed by a token with tag **I**. IOE2 is a variant of IOE1 in which each final word of a chunk is tagged with **E** regardless of whether is followed by a token inside a chunk. The representation used in the test data for CoNLL-2000 data set is the IOB2 representation. As a result, all figures reported using the CoNLL-2000 evaluation are based on this representation. Obviously, evaluation on different representations can be much higher but this kind of evaluation is not comparable with evaluations done using other representations and does not provide us with any insight about the methods we are evaluating. The objective in [SV99] was to see if the representation could improve accuracy for a single model. In contrast, in this paper we explore the question of whether these different representations can provide information that can be exploited using voting.

3.2 The Start/End Representation

This representation is often referred to as O+C and uses five tags. It was introduced in [UMM$^+$00]. This representation was used as a single representation in [KM01] but was not used in the voting scheme in that paper.

B Current token begins a chunk consisting of more than one token (think of it as [)
E Current token ends a chunk consisting of more than one token (think of it as])
I Current token is inside a chunk consisting of more than one token
S Current token is a chunk with exactly one token (think of it as [])
O Current token is outside any chunk

[SV99] explored the use of open and close brackets as well, but as separate un-coordinated representations, and hence without the need for the **S** tag above.

4 The Model

We wanted to use a simple machine learning model in order to discover the power of voting when applied to multiple data representations. We chose a simple trigram-based model trained using the maximum likelihood estimates based on frequencies from training data where the output label sequences (state transitions, in this case) are fully observed. Decoding on the test data set is done using the Viterbi algorithm[6]. In our experiments, we used the TnT tagger [Bra00] which implements the model described above. The trigram model used in this paper deals with unseen events by using linear interpolation. The trigram probability is interpolated with bigram and unigram probabilities, and the interpolation weights are chosen based on n-gram frequencies in training data [Bra00].

4.1 Baseline

The baseline results of the CoNLL-2000 and Base NP chunking were obtained by selecting the chunk tag which was most frequently associated with the current part-of-speech tag. The trigram model described above when using only part of speech tags as input. The model produced an $F_{\beta=1}$ score of 84.33 in IOB2 representation on CoNLL-2000 and an $F_{\beta=1}$ score of 79.99 in IOB1 representation on Base NP chunking as the output respectively.

4.2 Specialized Data Representation

Having only part of speech tags as input to the chunking model leads to low accuracy (as seen in the previous section). We use the proposals made in [MP02] to lexicalize our model by manipulating the data representation: changing the input and output of the function being learned to improve the accuracy of each learner. The key is to add

[6] Since we evaluate based on per word accuracy, a search for the best tag for each word, rather than the Viterbi best tag sequence will provide slightly higher accuracy. However, we use Viterbi since it is expedient and we focus on the improvement due to data representation voting.

Input			Output $= f_s$(Input)	
w_i	p_i	y_i	p_i or $w_i \cdot p_i$	$p_i \cdot y_i$ or $w_i \cdot p_i \cdot y_i$
You	PRP	B-NP	PRP	PRP-B-NP
will	MD	B-VP	MD	MD-B-VP
start	VB	I-VP	VB	VB-I-VP
to	TO	I-VP	TO	TO-I-VP
see	VB	I-VP	VB	VB-I-VP
shows	NNS	B-NP	NNS	NNS-B-NP
where	WRB	B-ADVP	where-WRB	where-WRB-B-ADVP
viewers	NNS	B-NP	NNS	NNS-B-NP
program	VBP	B-VP	VBP	VBP-B-VP
the	DT	B-NP	the-DT	the-DT-B-NP
program	NN	I-NP	NN	NN-I-NP

Fig. 2. Example of specialization where the words *where* and *the* belong to the set W_s.

lexicalization to the model, but to effectively deal with the sparse data problem. [MP02] propose the notion of *specialization* to deal with this issue. A *specialization function* converts the original input/output representation into a new representation which can then be used to train any model that could be trained on the original representation. The specialization function as defined by [MP02] closely fits into our framework of multiple data representations. The proposal is to produce a selective lexicalization of the original model by transforming the original data representation in the training data. The test data set is also transformed but we convert back to the original form before the evaluation step. Consider the original data which is a set of examples $L = \{\ldots, \langle w_i \cdot p_i\ ,\ y_i \rangle, \ldots\}$ where w_i is the input word, p_i is the input part of speech tag and y_i is the output chunk label (which varies depending on the representation used). A specialization function f_s uses a set of so-called *relevant* words W_s to convert each labelled example in L into a new representation in the following manner:

$$f_s(\langle w_i \cdot p_i\ ,\ y_i \rangle) = \begin{cases} \langle w_i \cdot p_i\ ,\ w_i \cdot p_i \cdot y_i \rangle & \text{if } w_i \in W_s \\ \langle p_i\ ,\ p_i \cdot y_i \rangle & \text{otherwise} \end{cases}$$

An example of specialization applied to the training data set is shown in Figure 2. We call the model **SP** if we only specialize the output using the part of speech tag (the *otherwise* case above). The selection of the set W_s produces various kinds of lexicalized models. Following [MP02] we use a development set consisting of a held-out or deleted set of 10% from the training set in order to pick elements for W_s. The held-out set consists of every 10th sentence. The remaining set is used as the training data. We used the following specialization representations. Each of these were defined in [MP02] and for each representation below also use the particular thresholds based on experiments with a held-out set or based on experiments on the training set:

SP+Lex-WHF W_s contains words whose frequency in the training set is higher than some threshold. The threshold is picked by testing on the held-out set. The threshold obtained in our experiments was 100.

Specialization criteria	P(%)	R(%)	$F_{\beta=1}$
Baseline	72.58	82.14	77.07
Trigram Model (no words)	84.31	84.35	84.33
SP	89.57	89.54	89.56
SP+Lex-WCH+WTE [MP02]	91.96	92.41	92.19
SP+Lex-WCH (5DR, Majority)	93.89	94.12	**94.01**
SP+Lex-WCH (3DR, Majority)	93.54	92.97	93.25
SP+Lex-WTE (3DR, Majority)	92.49	93.00	92.75

Table 1. Arbitrary phrase identification results for each setting.

Chunk type	P(%)	R(%)	$F_{\beta=1}$
ADJP	75.54	71.92	73.68
ADVP	80.80	79.21	80.00
CONJP	60.00	66.67	63.16
INTJ	50.00	50.00	50.00
NP	95.46	95.67	95.57
PP	97.69	96.61	97.15
PRT	66.02	64.15	65.07
SBAR	77.25	85.05	80.96
VP	92.69	94.16	93.42
all	93.89	94.12	**94.01**

Table 2. Arbitrary phrase identification results of 5DR majority voting with SP+Lex-WCH in IOB2.

SP+Lex-WCH W_s contains words that belong to certain chunk types *and* which are higher than some frequency threshold. In our experiments we pick chunk types NP, VP, PP and ADVP (the most frequent chunk types) with a threshold of 50.

SP+Lex-WTE W_s contains the words whose chunk tagging error rate was higher than some threshold on the held-out set. Based on the experiments in [MP02] we pick a threshold of 2.

The experiments in [MP02] show that specialization can improve performance considerably. By combining the **Lex-WCH** and **Lex-WTE** conditions, the output tag set increases from the original set of 22 to 1341, with 225 words being used as lexical material in the model and the accuracy on the CoNLL-2000 data increases to 92.19%[7] (see Table 1 for a comparison with our voting methods).

4.3 Voting between Multiple Representations

The notion of specialization is a good example of how the data representation can lead to higher accuracy. We extend this idea further by voting between multiple specialized data representations. The model we evaluate in this paper is simple majority voting on the output of various specialized trigram models (described above). The trigram model is trained on different data representations, and the test data set is decoded by each model. The output on the test data set is converted into a single representation, and the final label on the test data set is produced by a majority vote. We experimented with various weighted voting schemes, including setting weights for different representations based on accuracy on the held-out set, but no weighting scheme provided us with an increase in accuracy over simple majority voting. To save space, we only discuss majority voting in this paper[8].

[7] We replicated these results using the same trigram model described earlier.

[8] In future work, we plan to explore more sophisticated weighted voting schemes that exploit loss functions, e.g. AdaBoost.

Chunk type	P(%)	R(%)	$F_{\beta=1}$
ADJP	77.94	71.00	74.31
ADVP	80.12	78.18	79.14
CONJP	66.67	66.67	66.67
INTJ	50.00	50.00	50.00
NP	94.85	94.03	94.44
PP	97.52	96.47	96.99
PRT	64.29	59.43	61.76
SBAR	76.11	83.36	79.57
VP	92.65	93.39	93.02
all	93.54	92.97	93.25

Table 3. Arbitrary phrase identification results of 3DR majority voting with SP+Lex-WCH in IOB2.

	IOB1	IOB2	IOE1	IOE2	O+C
IOB1	92.68	93.07	92.66	92.68	94.72
IOB2	92.82	92.63	92.82	92.82	94.47
IOE1	92.82	92.82	92.87	92.87	94.64
IOE2	92.53	92.53	92.53	92.53	94.43
O+C	92.45	92.45	92.49	92.35	94.28
3DR	93.03	93.25	92.82	93.07	94.92
5DR	93.92	93.76	93.90	94.01	95.05

Table 4. Arbitrary phrase identification accuracy for all DRs in five evaluation formats. Each column represents the evaluation format and each row represents the training and testing format.

5 Experimental Results

In order to obtain various data representations, we transform the corpus[9] into the other data representations: IOB1(or IOB2), IOE1, IOE2 and O+C. We then transform each data representation into the format defined by specialized trigram model. We obtain a held-out set by splitting the original training set into a new training set (90% of the original training set) and a held-out set (10% of the original training set) for each data representation. Once we have all five different data representation chunked, we start to use majority voting technique to combine them into one file. In order to evaluate the accuracy, we have to transform the results into the evaluation format. This is trivial since all we need to do is to remove the enriched POS tag and lexical information from the specialized output file. In the results shown in this section, SP represents the specialized trigram model without lexical information; SP+Lex-WCH represents the specialized trigram model with lexical information defined based on Lex-WCH; 5DR represents five data representations (DR), which are IOB1, IOB2, IOE1, IOE2, O+C and we pick O+C as the default DR; 3DR represents IOB1, IOB2, IOE1 and we pick IOB2 as the default DR; Majority represents majority voting.

5.1 Arbitrary Phrase Identification (CoNLL-2000)

Table 1 gives the results of our specialized trigram model without lexical information. Based on these experiments, we select SP+Lex-WCH as our specialization technique for the voting experiments[10]. Table 2 and 3 show the results of our specialized trigram model with lexical information defined by Lex-WCH, where *all* represents the results

[9] CoNLL-2000 data set is originally in IOB2 format and Base NP data set is originally in IOB1 format.

[10] As pointed out by an anonymous reviewer, the number of representations participating in voting can be increased by considering all the specialization techniques. Further experiments are required, but these representations overlap substantially and so the gain is likely to be minimal.

obtained after 3DR or 5DR majority voting respectively. Table 6 compares the results with other major approaches. Tables 5 and 6 give the final results in IOB2 and IOE2 respectively. We achieved 94.01 on F-score for both formats, which is slightly higher than [KM01], but still lower than [ZDJ02] in Table 7. However, [ZDJ02] uses a full parser which we do not use in our experiments[11].

Voting format	P(%)	R(%)	$F_{\beta=1}$
IOB1	93.89	93.95	93.92
IOB2	93.69	93.82	93.76
IOE1	93.79	93.77	93.78
IOE2	93.89	94.12	94.01
O+C	93.84	93.98	93.91

Table 5. Arbitrary phrase identification accuracy for all DRs and all DRs are evaluated in IOB2. The voting format is the format when conducting majority voting, all the DRs are converted into this format.

Voting format	P(%)	R(%)	$F_{\beta=1}$
IOB1	93.81	93.79	93.80
IOB2	93.69	93.82	93.76
IOE1	93.87	93.93	93.90
IOE2	93.89	94.12	94.01
O+C	93.84	94.00	93.92

Table 6. Arbitrary phrase identification accuracy for all DRs evaluated in IOE1.

5.2 Base NP Identification

Table 9 shows the final results in IOB1 format after 5DR voting. Some published papers have picked IOB2 as their evaluation format. In our testing we have found no significant difference between IOB1 and IOB2 in terms of the results obtained, however, we will follow the IOB1 format since the original test data set is in IOB1 format.

Table 8 compares the Base NP chunking results with other major approaches. We achieved 95.23 on % $F_{\beta=1}$ score, which is the best state-of-the-art score so far. We also find the NP $F_{\beta=1}$ score obtained from an arbitrary phrase chunking process is slightly higher than that from a standard Base NP chunking process. This is to be expected since the arbitrary phrase chunking problem introduces additional constraints (since it is a multi-class model) when compared to the base NP chunker.

5.3 Runtime Performance

We compare our runtime performance against [KM01] since their accuracy is identical to ours in the CoNLL-2000 task (according to our significance tests). Our approach uses a simpler learner based on specialized trigram model, which runs in linear time, while [KM01] trains pairwise classifiers to reduce multi-class classification to binary classification, and they also apply weighted voting against multiple data representations.

[11] Our preliminary experiments along these lines using chunks obtained from a full parser have not yet produced an improvement in accuracy.

Approach	$F_{\beta=1}$
Generalized Winnow [ZDJ02] (with full parser)	94.17
Specialized Trigram Model w/ voting	**94.01**
SVM w/ DR voting [KM01]	93.90
Generalized Winnow [ZDJ02] (without full parser)	93.57
Voting w/ system combination [vH00]	93.32
MBL w/ multiple DR and system combination [San02]	92.50
Specialized Trigram Model [MP02] (no voting)	92.19

Table 7. Comparison of accuracy with major approaches for arbitrary phrase identification.

Approach	$F_{\beta=1}$
Specialized Trigram Model w/ voting	**95.23**
SVM w/ voting [KM01]	94.22
MBL w/ system combination [SDD^{+}00]	93.86
MBL w/ system combination [San02]	93.26

Table 8. Comparison of Base NP identification accuracy with major approaches.

Voting format	P(%)	R(%)	$F_{\beta=1}$
IOB1	95.11	95.35	**95.23**
IOB2	95.05	95.34	95.19
IOE1	94.96	95.11	95.04
IOE2	94.96	95.21	95.08
O+C	95.04	95.30	95.17

Table 9. Base NP identification accuracy for all DRs evaluated in IOB1 format. In the best performing case, IOB1 is the representation used for voting *and* the evaluation is in the IOB1 format.

Task	P-Value
CoNLL-2000	0.0745
Base NP	< 0.001

Table 10. McNemar's test between Specialized Trigram Model w/ voting and [KM01] on two chunking tasks.

Each SVM training step uses a quadratic programming step. Our simple voting system is considerably faster. Figure 3 compares the run time for training and decoding (testing) with a single data representation using our system compared with the time taken for the same task using the system of [KM01] (we downloaded their system and ran it on the same machine as our system).

5.4 Significance Testing

To examine the validity of the assumption that our approach is significantly different from that of [KM01] we applied the McNemar significance test (we assume the errors are made independently). The McNemar test showed that, for the CoNLL-2000 shared task of arbitrary phrase chunking, our score is not statistically significantly better than the results in [KM01]. However for the Base NP chunking task, our score is indeed better by a statistically significant margin from the results in [KM01].

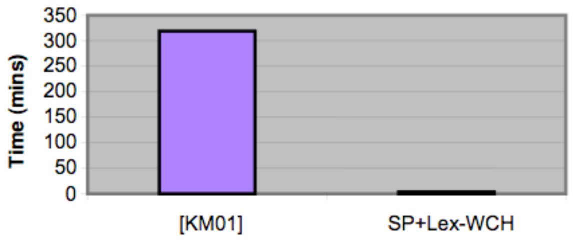

Fig. 3. Comparison of run times for training from the training data and decoding the test data between our system and that of [KM01] for a single data representation.

6 Comparison with Other Voting Methods

We apply simple majority voting between five data representations (Inside/Outside and Start/End), while [KM01] only apply weighted voting between Inside/Outside representations, since their learner restricted them to vote between different data representation types. In our experiments, we find the Start/End representation usually catches more information than the Inside/Outside representations and in turn improves our performance. Previous approaches that use voting have all used voting as a means of system combination, i.e. taking multiple machine learning methods and taking a majority vote or weighted vote combining their output [SDD$^+$00]. This kind of system combination can be done using voting or stacking. Voting as system combination has been applied to the CoNLL-2000 data set as well: [vH00] obtains an $F_{\beta=1}$ of 93.32. [San02] combines the output of several systems but also does voting by exploiting different data representations. However, to our knowledge, there has not been a study of voting purely between multiple data representations using a single machine learning method ([San00] is a study of voting between multiple data representations but it combines this approach with multiple pass chunking in the same experimental study). Our results seem to indicate that even simple majority voting between multiple data representations does better than voting as a means for system combination.

7 Conclusion

The main contribution of this paper is that a single learning method, a simple trigram model can use voting between multiple data representations to obtain results equal to the best on the CoNLL-2000 text chunking data set. Using no additional knowledge sources, we achieved 94.01% $F_{\beta=1}$ score compared to the previous best comparable score of 93.90% and an $F_{\beta=1}$ score of 95.23% on the Base NP identification task compared to the previous best comparable score of 94.22%. Using the McNemar significance test, we show that our results is significantly better than the current comparable state-of-the-art approach on the Base NP chunking task. In addition, our text chunker is faster by several orders of magnitude than comparably accurate methods. We are faster both in time taken in training as well as decoding.

In our experiments, we use simple majority voting because we found weighted voting to be less accurate in our case: we use only five data representations (DRs) that were chosen carefully each with high accuracy. Weighted voting will be particularly useful if we scale up the number of distinct DRs, some which may not accurate overall but provide correct answers for certain difficult examples. Many DRs could potentially be created if we encode the context into the DR, e.g. a tag could be **IO** if the current tag is **I** and the previous tag is **O**. Other DRs could use tags that encode syntactic structure, e.g. SuperTagging [Sri97] uses a tagging model to provide a piece of syntactic structure to each word in the input and can be used as a DR that can participate in voting. In each case, an efficient mapping from the DR into a common voting format will be needed.

References

[Bra00] T. Brants. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied Natural Language Processing Conference: ANLP-2000*, Seattle, USA, 2000.

[Cha96] E. Charniak. *Statistical Language Learning*. MIT Press, 1996.

[KM01] T. Kudo and Y. Matsumoto. Chunking with support vector machines. In *Proceedings of the 2nd Meeting of the North American Association for Computational Linguistics: NAACL 2001*, 2001.

[MP02] A. Molina and F. Pla. Shallow Parsing using Specialized HMMs. *Journal of Machine Learning Research*, 2:595–613, March 2002.

[RM95] L. Ramshaw and M. Marcus. Text Chunking using Transformation-Based Learning. In *Proceedings of the 3rd Workshop on Very Large Corpora: WVLC-1995*, Cambridge, USA, 1995.

[San00] E. F. Tjong Kim Sang. Text chunking by system combination. In *Proceedings of the Conference on Computational Natural Language Learning: CoNLL-2000*, pages 151–153, Lisbon, Portugal, 2000.

[San02] E. F. Tjong Kim Sang. Memory-based shallow parsing. *Journal of Machine Learning Research*, 2:559–594, March 2002.

[SB00] E. F. Tjong Kim Sang and S. Buchholz. Introduction to the CoNLL-2000 Shared Task: Chunking. In *Proceedings of the Conference on Computational Natural Language Learning: CoNLL-2000*, pages 151–153, Lisbon, Portugal, 2000.

[SDD$^+$00] E. F. Tjong Kim Sang, Walter Daelemans, Hervé Déjean, Rob Koeling, Yuval Krymolowski, Vasin Punyakanok, and Dan Roth. Applying system combination to base noun phrase identification. In *Proceedings of COLING-2000*, pages 857–863, Saarbuecken, Germany, 2000.

[Sri97] B. Srinivas. Performance evaluation of supertagging for partial parsing. In *Proc. of Fifth International Workshop on Parsing Technologies*, Boston, September 1997.

[SV99] E. F. Tjong Kim Sang and J. Veenstra. Representing texting chunks. In *Proceedings of the 7th Conference of the European Association for Computational Linguistics: EACL-1999*, pages 173–179, Bergen, Norway, 1999.

[UMM$^+$00] K. Uchimoto, Q. Ma, M. Murata, H. Ozaku, and H. Isahara. Named Entity Extraction based on a Maximum Entropy Model and Transformation Rules. In *Proceedings of the 38th Meeting of the ACL: ACL-2000*, 2000.

[vH00] Hans van Halteren. Chunking with WPDV Models. In *Proceedings of the Conference on Computational Natural Language Learning: CoNLL-2000*, pages 154–156, Lisbon, Portugal, 2000.

[ZDJ02] T. Zhang, F. Damerau, and D. Johnson. Text Chunking based on a Generalization of Winnow. *Journal of Machine Learning Research*, 2:615–637, March 2002.