Applying System Combination to Base Noun Phrase Identification

Erik F. Tjong Kim Sang[α], Walter Daelemans[α], Hervé Déjean[τ],
Rob Koeling[γ], Yuval Krymolowski[β], Vasin Punyakanok[ι], Dan Roth[ι]

[α]University of Antwerp
Universiteitsplein 1
B-2610 Wilrijk, Belgium
{erikt,daelem}@uia.ua.ac.be

[τ]Universität Tübingen
Kleine Wilhelmstraße 113
D-72074 Tübingen, Germany
dejean@sfs.nphil.uni-tuebingen.de

[γ]SRI Cambridge
23 Millers Yard,Mill Lane
Cambridge, CB2 1RQ, UK
koeling@cam.sri.com

[β]Bar-Ilan University
Ramat Gan, 52900, Israel
yuvalk@macs.biu.ac.il

[ι]University of Illinois
1304 W. Springfield Ave.
Urbana, IL 61801, USA
{punyakan,danr}@cs.uiuc.edu

Abstract

We use seven machine learning algorithms for one task: identifying base noun phrases. The results have been processed by different system combination methods and all of these outperformed the best individual result. We have applied the seven learners with the best combinator, a majority vote of the top five systems, to a standard data set and managed to improve the best published result for this data set.

1 Introduction

Van Halteren et al. (1998) and Brill and Wu (1998) show that part-of-speech tagger performance can be improved by combining different taggers. By using techniques such as majority voting, errors made by the minority of the taggers can be removed. Van Halteren et al. (1998) report that the results of such a combined approach can improve upon the accuracy error of the best individual system with as much as 19%. The positive effect of system combination for non-language processing tasks has been shown in a large body of machine learning work.

In this paper we will use system combination for identifying base noun phrases (baseNPs). We will apply seven machine learning algorithms to the same baseNP task. At two points we will apply combination methods. We will start with making the systems process five output representations and combine the results by choosing the majority of the output features. Three of the seven systems use this approach. After this we will make an overall combination of the results of the seven systems. There we will evaluate several system combination methods. The best performing method will be applied to a standard data set for baseNP identification.

2 Methods and experiments

In this section we will describe our learning task: recognizing base noun phrases. After this we will describe the data representations we used and the machine learning algorithms that we will apply to the task. We will conclude with an overview of the combination methods that we will test.

2.1 Task description

Base noun phrases (baseNPs) are noun phrases which do not contain another noun phrase. For example, the sentence

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

contains six baseNPs (marked as phrases between square brackets). The phrase *$ 366.50 an ounce* is a noun phrase as well. However, it is not a baseNP since it contains two other noun phrases. Two baseNP data sets have been put forward by Ramshaw and Marcus (1995). The main data set consist of four sections of the Wall Street Journal (WSJ) part of the Penn Treebank (Marcus et al., 1993) as training material (sections 15-18, 211727 tokens) and one section as test material (section 20, 47377 tokens)[1]. The data contains words, their part-of-speech

[1] This Ramshaw and Marcus (1995) baseNP data set is available via ftp://ftp.cis.upenn.edu/pub/chunker/

(POS) tags as computed by the Brill tagger and their baseNP segmentation as derived from the Treebank (with some modifications).

In the baseNP identification task, performance is measured with three rates. First, with the percentage of detected noun phrases that are correct (precision). Second, with the percentage of noun phrases in the data that were found by the classifier (recall). And third, with the $F_{\beta=1}$ rate which is equal to (2*precision*recall)/(precision+recall). The latter rate has been used as the target for optimization.

2.2 Data representation

In our example sentence in section 2.1, noun phrases are represented by bracket structures. It has been shown by Muñoz et al. (1999) that for baseNP recognition, the representation with brackets outperforms other data representations. One classifier can be trained to recognize open brackets (O) and another can handle close brackets (C). Their results can be combined by making pairs of open and close brackets with large probability scores. We have used this bracket representation (O+C) as well. However, we have not used the combination strategy from Muñoz et al. (1999) but instead used the strategy outlined in Tjong Kim Sang (2000): regard only the shortest possible phrases between candidate open and close brackets as base noun phrases.

An alternative representation for baseNPs has been put forward by Ramshaw and Marcus (1995). They have defined baseNP recognition as a tagging task: words can be inside a baseNP (I) or outside a baseNP (O). In the case that one baseNP immediately follows another baseNP, the first word in the second baseNP receives tag B. Example:

> In$_O$ early$_I$ trading$_I$ in$_O$ Hong$_I$ Kong$_I$
> Monday$_B$,$_O$ gold$_I$ was$_O$ quoted$_O$ at$_O$
> \$$_I$ 366.50$_I$ an$_B$ ounce$_I$.$_O$

This set of three tags is sufficient for encoding baseNP structures since these structures are nonrecursive and nonoverlapping.

Tjong Kim Sang (2000) outlines alternative versions of this tagging representation. First, the B tag can be used for the first word of every baseNP (IOB2 representation). Second, instead of the B tag an E tag can be used to mark the last word of a baseNP immediately before another baseNP (IOE1). And third, the E tag can be used for every noun phrase final word (IOE2). He used the Ramshaw and Marcus (1995) representation as well (IOB1). We will use these four tagging representations and the O+C representation for the system-internal combination experiments.

2.3 Machine learning algorithms

This section contains a brief description of the seven machine learning algorithms that we will apply to the baseNP identification task: ALLiS, C5.0, IGTree, MaxEnt, MBL, MBSL and SNoW.

ALLiS[2] (Architecture for Learning Linguistic Structures) is a learning system which uses theory refinement in order to learn non-recursive NP and VP structures (Déjean, 2000). ALLiS generates a regular expression grammar which describes the phrase structure (NP or VP). This grammar is then used by the CASS parser (Abney, 1996). Following the principle of theory refinement, the learning task is composed of two steps. The first step is the generation of an *initial grammar*. The generation of this grammar uses the notion of default values and some background knowledge which provides general expectations concerning the inner structure of NPs and VPs. This initial grammar provides an incomplete and/or incorrect analysis of the data. The second step is the refinement of this grammar. During this step, the validity of the rules of the initial grammar is checked and the rules are improved (refined) if necessary. This refinement relies on the use of two operations: the contextualization (in which contexts such a tag always belongs to the phrase) and lexicalization (use of information about the words and not only about POS).

C5.0[3], a commercial version of C4.5 (Quinlan, 1993), performs *top-down induction of decision trees* (TDIDT). On the basis of an instance base of examples, C5.0 constructs a decision tree which compresses the classification information in the instance base by exploiting differences in relative importance of different features. Instances are stored in the tree as paths

[2] A demo of the NP and VP chunker is available at http://www.sfb441.unituebingen.de/~dejean/chunker.html

[3] Available from http://www.rulequest.com

of connected nodes ending in leaves which contain classification information. Nodes are connected via arcs denoting feature values. Feature information gain (mutual information between features and class) is used to determine the order in which features are employed as tests at all levels of the tree (Quinlan, 1993). With the full input representation (words and POS tags), we were not able to run complete experiments. We therefore experimented only with the POS tags (with a context of two left and right). We have used the default parameter setting with decision trees combined with value grouping.

We have used a nearest neighbor algorithm (IB1-IG, here listed as MBL) and a decision tree algorithm (IGTree) from the TiMBL learning package (Daelemans et al., 1999b). Both algorithms store the training data and classify new items by choosing the most frequent classification among training items which are closest to this new item. Data items are represented as sets of feature-value pairs. Each feature receives a weight which is based on the amount of information which it provides for computing the classification of the items in the training data. IB1-IG uses these weights for computing the distance between a pair of data items and IGTree uses them for deciding which feature-value decisions should be made in the top nodes of the decision tree (Daelemans et al., 1999b). We will use their default parameters except for the IB1-IG parameter for the number of examined nearest neighbors (k) which we have set to 3 (Daelemans et al., 1999a). The classifiers use a left and right context of four words and part-of-speech tags. For the four IO representations we have used a second processing stage which used a smaller context but which included information about the IO tags predicted by the first processing phase (Tjong Kim Sang, 2000).

When building a classifier, one must gather evidence for predicting the correct class of an item from its context. The Maximum Entropy (MaxEnt) framework is especially suited for integrating evidence from various information sources. Frequencies of evidence/class combinations (called features) are extracted from a sample corpus and considered to be properties of the classification process. Attention is constrained to models with these properties. The MaxEnt principle now demands that among all the probability distributions that obey these constraints, the most uniform is chosen. During training, features are assigned weights in such a way that, given the MaxEnt principle, the training data is matched as well as possible. During evaluation it is tested which features are *active* (i.e. a feature is active when the context meets the requirements given by the feature). For every class the weights of the active features are combined and the best scoring class is chosen (Berger et al., 1996). For the classifier built here the surrounding words, their POS tags and baseNP tags predicted for the previous words are used as evidence. A mixture of simple features (consisting of one of the mentioned information sources) and complex features (combinations thereof) were used. The left context never exceeded 3 words, the right context was maximally 2 words. The model was calculated using existing software (Dehaspe, 1997).

MBSL (Argamon et al., 1999) uses POS data in order to identify baseNPs. Inference relies on a memory which contains all the occurrences of POS sequences which appear in the beginning, or the end, of a baseNP (including complete phrases). These sequences may include a few context tags, up to a prespecified *max_context*. During inference, MBSL tries to 'tile' each POS string with parts of noun-phrases from the memory. If the string could be fully covered by the tiles, it becomes part of a candidate list, ambiguities between candidates are resolved by a constraint propagation algorithm. Adding a context extends the possibilities for tiling, thereby giving more opportunities to better candidates. The approach of MBSL to the problem of identifying baseNPs is sequence-based rather than word-based, that is, decisions are taken per POS sequence, or per candidate, but not for a single word. In addition, the tiling process gives no preference to any direction in the sentence. The tiles may be of any length, up to the maximal length of a phrase in the training data, which gives MBSL a generalization power that compensates for the setup of using only POS tags. The results presented here were obtained by optimizing MBSL parameters based on 5-fold CV on the training data.

SNoW uses the Open/Close model, described in Muñoz et al. (1999). As is shown there, this

	MBL			MaxEnt			IGTree		
section 21	O	C	$F_{\beta=1}$	O	C	$F_{\beta=1}$	O	C	$F_{\beta=1}$
IOB1	97.81%	97.97%	91.68	97.90%	98.11%	92.43	96.62%	96.89%	87.88
IOB2	97.63%	97.96%	91.79	97.81%	98.14%	92.14	97.27%	97.30%	90.03
IOE1	97.80%	97.92%	91.54	97.88%	98.12%	92.37	95.88%	96.01%	82.80
IOE2	97.72%	97.94%	92.06	97.84%	98.12%	92.13	97.19%	97.62%	89.98
O+C	97.72%	98.04%	92.03	97.82%	98.15%	92.26	96.89%	97.49%	89.37
Majority	98.04%	98.20%	92.82	97.94%	98.24%	92.60	97.70%	97.99%	91.92

Table 1: The effects of system-internal combination by using different output representations. A straight-forward majority vote of the output yields better bracket accuracies and $F_{\beta=1}$ rates than any included individual classifier. The bracket accuracies in the columns O and C show what percentage of words was correctly classified as baseNP start, baseNP end or neither.

model produced better results than the other paradigm evaluated there, the Inside/Outside paradigm. The Open/Close model consists of two SNoW predictors, one of which predicts the beginning of baseNPs (Open predictor), and the other predicts the end of the phrase (Close predictor). The Open predictor is learned using SNoW (Carlson et al., 1999; Roth, 1998) as a function of features that utilize words and POS tags in the sentence and, given a new sentence, will predict for each word whether it is the first word in the phrase or not. For each Open, the Close predictor is learned using SNoW as a function of features that utilize the words in the sentence, the POS tags and the open prediction. It will predict, for each word, whether it can be the end of the phrase, given the previously predicted Open. Each pair of predicted Open and Close forms a candidate of a baseNP. These candidates may conflict due to overlapping; at this stage, a graph-based constraint satisfaction algorithm that uses the confidence values SNoW associates with its predictions is employed. This algorithm ("the combinator") produces the list of the final baseNPs for each sentence. Details of SNoW, its application in shallow parsing and the combinator's algorithm are in Muñoz et al. (1999).

2.4 Combination techniques

At two points in our noun phrase recognition process we will use system combination. We will start with system-internal combination: apply the same learning algorithm to variants of the task and combine the results. The approach we have chosen here is the same as in Tjong Kim Sang (2000): generate different variants of the task by using different representations of the output (IOB1, IOB2, IOE1, IOE2 and O+C). The five outputs will converted to the open bracket representation (O) and the close bracket representation (C) and after this, the most frequent of the five analyses of each word will chosen (majority voting, see below). We expect the systems which use this combination phase to perform better than their individual members (Tjong Kim Sang, 2000).

Our seven learners will generate different classifications of the training data and we need to find out which combination techniques are most appropriate. For the system-external combination experiment, we have evaluated different voting mechanisms, effectively the voting methods as described in Van Halteren et al. (1998). In the first method each classification receives the same weight and the most frequent classification is chosen (Majority). The second method regards as the weight of each individual classification algorithm its accuracy on some part of the data, the tuning data (TotPrecision). The third voting method computes the precision of each assigned tag per classifier and uses this value as a weight for the classifier in those cases that it chooses the tag (TagPrecision). The fourth method uses both the precision of each assigned tag and the recall of the competing tags (Precision-Recall). Finally, the fifth method uses not only a weight for the current classification but it also computes weights for other possible classifications. The other classifications are determined by examining the tuning data and registering the correct values for every pair of classifier results (pair-wise voting, see Van Halteren et al. (1998) for an elaborate explanation).

Apart from these five voting methods we have also processed the output streams with two classifiers: MBL and IGTree. This approach is called classifier stacking. Like Van Halteren et al. (1998), we have used different input versions: one containing only the classifier output and another containing both classifier output and a compressed representation of the data item under consideration. For the latter purpose we have used the part-of-speech tag of the current word.

3 Results[4]

We want to find out whether system combination could improve performance of baseNP recognition and, if this is the fact, we want to select the best combination technique. For this purpose we have performed an experiment with sections 15-18 of the WSJ part of the Penn Treebank as training data (211727 tokens) and section 21 as test data (40039 tokens). Like the data used by Ramshaw and Marcus (1995), this data was retagged by the Brill tagger in order to obtain realistic part-of-speech (POS) tags[5]. The data was segmented into baseNP parts and non-baseNP parts in a similar fashion as the data used by Ramshaw and Marcus (1995). Of the training data, only 90% was used for training. The remaining 10% was used as tuning data for determining the weights of the combination techniques.

For three classifiers (MBL, MaxEnt and IGTree) we have used system-internal combination. These learning algorithms have processed five different representations of the output (IOB1, IOB2, IOE1, IOE2 and O+C) and the results have been combined with majority voting. The test data results can be found in Table 1. In all cases, the combined results were better than that of the best included system.

The results of ALLiS, C5.0, MBSL and SNoW have been converted to the O and the C representation. Together with the bracket representations of the other three techniques, this gave us a total of seven O results and seven C results. These two data streams have been combined with the combination techniques described in section 2.4. After this, we built baseNPs from the O and C results of each combination technique, like described in section 2.2. The bracket accuracies and the $F_{\beta=1}$ scores for test data can be found in Table 2.

section 21	O	C	$F_{\beta=1}$
Classifier			
ALLiS	97.87%	98.08%	92.15
C5.0	97.05%	97.76%	89.97
IGTree	97.70%	97.99%	91.92
MaxEnt	97.94%	98.24%	92.60
MBL	98.04%	98.20%	92.82
MBSL	97.27%	97.66%	90.71
SNoW	97.78%	97.68%	91.87
Simple Voting			
Majority	98.08%	98.21%	92.95
TotPrecision	98.08%	98.21%	92.95
TagPrecision	98.08%	98.21%	92.95
Precision-Recall	98.08%	98.21%	92.95
Pairwise Voting			
TagPair	98.13%	98.23%	93.07
Memory-Based			
Tags	98.24%	98.35%	93.39
Tags + POS	98.14%	98.33%	93.24
Decision Trees			
Tags	98.24%	98.35%	93.39
Tags + POS	98.13%	98.32%	93.21

Table 2: Bracket accuracies and $F_{\beta=1}$ scores for section WSJ 21 of the Penn Treebank with seven individual classifiers and combinations of them. Each combination performs better than its best individual member. The stacked classifiers without context information perform best.

All combinations improve the results of the best individual classifier. The best results were obtained with a memory-based stacked classifier. This is different from the combination results presented in Van Halteren et al. (1998), in which pairwise voting performed best. However, in their later work stacked classifiers outperform voting methods as well (Van Halteren et al., to appear).

[4] Detailed results of our experiments are available on http://lcg-www.uia.ac.be/~erikt/npcombi/

[5] The retagging was necessary to assure that the performance rates obtained here would be similar to rates obtained for texts for which no Treebank POS tags are available.

section 20	accuracy	precision	recall	$F_{\beta=1}$
Best-five combination	O:98.32% C:98.41%	94.18%	93.55%	93.86
Tjong Kim Sang (2000)	O:98.10% C:98.29%	93.63%	92.89%	93.26
Muñoz et al. (1999)	O:98.1% C:98.2%	92.4%	93.1%	92.8
Ramshaw and Marcus (1995)	IOB1:97.37%	91.80%	92.27%	92.03
Argamon et al. (1999)	-	91.6%	91.6%	91.6

Table 3: The overall performance of the majority voting combination of our best five systems (selected on tuning data performance) applied to the standard data set put forward by Ramshaw and Marcus (1995) together with an overview of earlier work. The accuracy scores indicate how often a word was classified correctly with the representation used (O, C or IOB1). The combined system outperforms all earlier reported results for this data set.

Based on an earlier combination study (Tjong Kim Sang, 2000) we had expected the voting methods to do better. We suspect that their performance is below that of the stacked classifiers because the difference between the best and the worst individual system is larger than in our earlier study. We assume that the voting methods might perform better if they were only applied to the classifiers that perform well on this task. In order to test this hypothesis, we have repeated the combination experiments with the best n classifiers, where n took values from 3 to 6 and the classifiers were ranked based on their performance on the tuning data. The best performances were obtained with five classifiers: $F_{\beta=1}$=93.44 for all five voting methods with the best stacked classifier reaching 93.24. With the top five classifiers, the voting methods outperform the best combination with seven systems[6]. Adding extra classification results to a good combination system should not make overall performance worse so it is clear that there is some room left for improvement of our combination algorithms.

We conclude that the best results in this task can be obtained with the simplest voting method, majority voting, applied to the best five of our classifiers. Our next task was to apply the combination approach to a standard data set so that we could compare our results with other work. For this purpose we have used the data put forward by Ramshaw and Marcus (1995). Again, only 90% of the training data was used for training while the remaining 10% was reserved for ranking the classifiers. The seven learners were trained with the same parameters as in the previous experiment. Three of the classifiers (MBL, MaxEnt and IGTree) used system-internal combination by processing different output representations.

The classifier output was converted to the O and the C representation. Based on the tuning data performance, the classifiers ALLiS, IGTREE, MaxEnt, MBL and SNoW were selected for being combined with majority voting. After this, the resulting O and C representations were combined to baseNPs by using the method described in section 2.2. The results can be found in Table 3. Our combined system obtains an $F_{\beta=1}$ score of 93.86 which corresponds to an 8% error reduction compared with the best published result for this data set (93.26).

4 Concluding remarks

In this paper we have examined two methods for combining the results of machine learning algorithms for identifying base noun phrases. In the first method, the learner processed different output data representations and the results were combined by majority voting. This approach yielded better results than the best included classifier. In the second combination approach we have combined the results of seven learning systems (ALLiS, C5.0, IGTree, MaxEnt, MBL, MBSL and SNoW). Here we have tested different combination methods. Each combination

[6] We are unaware of a good method for determining the significance of $F_{\beta=1}$ differences but we assume that this $F_{\beta=1}$ difference is not significant. However, we believe that the fact that more combination methods perform well, shows that it easier to get a good performance out of the best five systems than with all seven.

method outperformed the best individual learning algorithm and a majority vote of the top five systems performed best. We have applied this approach of system-internal and system-external combination to a standard data set for base noun phrase identification and the performance of our system was better than any other published result for this data set.

Our study shows that the combination methods that we have tested are sensitive for the inclusion of classifier results of poor quality. This leaves room for improvement of our results by evaluating other combinators. Another interesting approach which might lead to a better performance is taking into account more context information, for example by combining complete phrases instead of independent brackets. It would also be worthwhile to evaluate using more elaborate methods for building baseNPs out of open and close bracket candidates.

Acknowledgements

Déjean, Koeling and Tjong Kim Sang are funded by the TMR network Learning Computational Grammars[7]. Punyakanok and Roth are supported by NFS grants IIS-9801638 and SBR-9873450.

References

Steven Abney. 1996. Partial parsing via finite-state cascades. In *In Proceedings of the ESSLLI '96 Robust Parsing Workshop.*

Shlomo Argamon, Ido Dagan, and Yuval Krymolowski. 1999. A memory-based approach to learning shallow natural language patterns. *Journal of Experimental and Theoretical AI*, 11(3).

Adam L. Berger, Stephen A. DellaPietra, and Vincent J. DellaPietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1).

Eric Brill and Jun Wu. 1998. Classifier combination for improved lexical disambiguation. In *Proceedings of COLING-ACL '98.* Association for Computational Linguistics.

A. Carlson, C. Cumby, J. Rosen, and D. Roth. 1999. The SNoW learning architecture. Technical Report UIUCDCS-R-99-2101, UIUC Computer Science Department, May.

Walter Daelemans, Antal van den Bosch, and Jakub Zavrel. 1999a. Forgetting exceptions is harmful in language learning. *Machine Learning*, 34(1).

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 1999b. *TiMBL: Tilburg Memory Based Learner, version 2.0, Reference Guide.* ILK Technical Report 99-01. http://ilk.kub.nl/.

Luc Dehaspe. 1997. Maximum entropy modeling with clausal constraints. In *Proceedings of the 7th International Workshop on Inductive Logic Programming.*

Hervé Déjean. 2000. Theory refinement and natural language processing. In *Proceedings of the Coling2000.* Association for Computational Linguistics.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of english: the penn treebank. *Computational Linguistics*, 19(2).

Marcia Munoz, Vasin Punyakanok, Dan Roth, and Dav Zimak. 1999. A learning approach to shallow parsing. In *Proceedings of EMNLP-WVLC'99.* Association for Computational Linguistics.

J. Ross Quinlan. 1993. *c4.5: Programs for Machine Learning.* Morgan Kaufmann.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the Third ACL Workshop on Very Large Corpora.* Association for Computational Linguistics.

D. Roth. 1998. Learning to resolve natural language ambiguities: A unified approach. In *AAAI-98.*

Erik F. Tjong Kim Sang. 2000. Noun phrase recognition by system combination. In *Proceedings of the ANLP-NAACL-2000.* Seattle, Washington, USA. Morgan Kaufman Publishers.

Hans van Halteren, Jakub Zavrel, and Walter Daelemans. 1998. Improving data driven wordclass tagging by system combination. In *Proceedings of COLING-ACL '98.* Association for Computational Linguistics.

Hans van Halteren, Jakub Zavrel, and Walter Daelemans. to appear. Improving accuracy in nlp through combination of machine learning systems.

[7] http://lcg-www.uia.ac.be/