Memory-Based Clause Identification

Erik F. Tjong Kim Sang
CNTS - Language Technology Group
University of Antwerp
erikt@uia.ua.ac.be

1 Introduction

We apply a memory-based learner to the CoNLL-2001 shared task: clause identification (Tjong Kim Sang and Déjean, 2001). The task is divided in three parts. The first two parts are classification tasks: identifying the positions of clause starts and clause ends given a word, its part-of-speech tag and the syntactic base chunk it belongs to. Our memory-based learner can be applied to these tasks in a straightforward way. The third part of the shared task is identifying complete embedded clauses. We will perform this task by first identifying clause starts and clause ends and then combining these to clauses with a set of conversion rules.

2 Approach

The first two parts of the CoNLL-2001 shared task are similar to the CoNLL-2000 shared task: classify words in context according to some tagging scheme. We have participated in the latter shared task (Tjong Kim Sang, 2000) and we will use a similar approach for the first two parts of the 2001 shared task. The goal of these parts is to predict if a word is the first word of a clause or not (part 1) and if it is the final word of a clause or not (part 2). We have used the memory-based classifier TiMBL (Daelemans et al., 2000) for predicting the most likely classification of each word. Memory-based learners store all training data and determine a classification for new data by examining the classifications of training data which are similar to the new data. Each item is represented by a set of feature-value pairs. The features have weights which encode their relevance to the classification of the training data items (Daelemans et al., 2000).

Although the memory-based learner is able to find a sensible feature weight set, it is not guaranteed to find the best feature weight set. In an earlier study (Tjong Kim Sang and Veenstra, 1999), we have reported that the number of features supplied to the system has an influence on the performance and that the maximal number of features not necessarily provided the best results. In order to maximize the system's performance, we have evaluated seven combinations of the available three feature types (words, part-of-speech (POS) tags and clause tags):

1. words only (w)
2. POS tags only (p)
3. clause tags only (c)
4. words and POS tags (wp)
5. words and clause tags (wc)
6. POS tags and clause tags (pc)
7. words, POS tags and clause tags (wpc)

In our CoNLL-2000 work, we have shown that it is useful to evaluate combinations of classifiers since these often tend to perform better than their best individual member (Tjong Kim Sang, 2000). We will also use this approach here and will evaluate majority votes of the classifier combinations 1+2+3 (8), 4+5+6 (9) and 7+8+9 (10). This means, for example, that we will look at the output of the classifiers 1, 2 and 3 of the list above. Each of these classifiers predicts that a word starts a clause or not (task 1). We generate a new data classification (8) by choosing for each word the classification which is predicted most frequently.

The ten set-ups we have described above, use information about all words. However, clauses are a high-level structures which might not need all this information. It might be useful to replace the chunks by a single token since our fixed-context system might be able to make better judgements when it is able to examine a

	train1	0	1	2	3
1	w	61.77	84.40	83.74	81.08
2	p	30.44	80.40	80.47	76.85
3	c	13.67	76.76	79.05	78.71
4	wp	62.24	87.19	84.45	81.22
5	wc	67.95	87.31	85.74	82.97
6	pc	49.29	86.65	84.92	81.72
7	wpc	68.66	87.92	85.93	83.28
8	1+2+3	38.32	85.24	86.92	85.38
9	4+5+6	68.04	88.83	87.44	84.98
10	7+8+9	68.03	88.75	87.72	85.45
11	w-	54.05	83.70	83.48	81.25
12	c-	14.26	77.70	79.30	78.50
13	wc-	58.47	86.53	85.74	82.77

Table 1: $F_{\beta=1}$ rates obtained in 10-fold cross-validation experiments with the training data of part 1 of the shared task. We used different combinations of information (w: words, p: POS tags and c: chunk tags) and different context sizes (0-3). The best results have been obtained with a majority vote of three information pairs while using context size 1 (row 9).

	train2	0	1	2	3
1	w	61.11	75.99	77.52	77.63
2	p	61.71	77.52	78.74	77.95
3	c	00.00	67.25	75.06	75.70
4	wp	61.25	76.52	77.92	78.12
5	wc	61.01	75.96	77.46	77.79
6	pc	61.74	77.44	78.40	77.93
7	wpc	61.21	76.17	77.73	78.00
8	1+2+3	61.67	75.93	79.60	79.94
9	4+5+6	61.44	77.30	79.15	79.38
10	7+8+9	61.44	77.20	79.25	79.60
11	w-	61.24	76.01	78.69	79.25
12	c-	61.73	76.82	78.34	80.90
13	wc-	61.43	76.77	80.15	81.61

Table 2: $F_{\beta=1}$ rates obtained in 10-fold cross-validation experiments with the training data of part 2 of the shared task. We used different combinations of information (w: words, p: POS tags and c: chunk tags) and different context sizes (0-3). The best results have been obtained with words and POS tags after compressing the chunks and while using context size 3 (row 13).

larger part of a sentence. We have tested this by removing all chunks from the data and replacing them by their head word and the chunk tag. The head words have been generated by a set of rules put forward by (Magerman, 1995) and modified by (Collins, 1999)[1]. Words that are outside of a base chunk receive their POS tag as chunk tag. This approach consists of three feature combinations: words only (w-), chunk tags only (c-) and words and chunk tags (wc-).

The evaluation has been performed with 10-fold cross-validation on the training data to avoid tuning the system parameters on the test data. This means that we have split the training data in 10 parts and tested each of the parts after having trained with the other nine. The 10 results have been concatenated and processed by the evaluation software for the shared task.

In this evaluation process we have also tested different symmetric sizes of the context: 0, 1, 2 and 3. For example, while classifying the fourth word of the phrase *But analysts reckon underlying support for sterling*, we used only one feature for context size 0 (*underlying*) and five for context size 2 (*analysts, reckon, underlying, support, for*): the focus word and the two previous and the two next words. In our previous work we found that the performance increased for larger context sizes until some task-dependent size after which the performance dropped gradually (Tjong Kim Sang and Veenstra, 1999).

In principle, the third part of the shared task can also be interpreted as a classification task. However, the task is more difficult since besides predicting where clauses start and end, it requires as well predicting how many clauses start or end at a certain position. This is a sheer impossible task for a system which examines no more than 7 tokens at a time but needs to process sentences with an average length of more than 20 tokens. Rather than solving the impossible, we have tried to use the results of the other two parts of the shared task, start and end positions of clauses, for building a complete clause structure. For this purpose we have used the following heuristic rules:

1. Assume that exactly one clause starts at each clause start position.
2. Assume that exactly one clause ends at each clause end position but

[1]Available on http://www.research.att.com/~mcollins/papers/heads

3. ignore all clause end positions when currently no clause is open, and
4. ignore all clause ends at non-sentence-final positions which attempt to close a clause started at the first word of the sentence.
5. If clauses are opened but not closed at the end of the sentence then close them at the penultimate word of the sentence.

These rules generate complete and consistent embedded clause structures from the clause boundary output of our experiments.

3 Results

We have performed the evaluations described in the previous section in a 10-fold cross-validation experiment on the training data of parts 1 and 2 of the shared task. The results can be found in tables 1 and 2. For the clause start prediction part (1), we obtained the best performance with a majority vote of the results for the feature combinations wp, wc and pc (row 9) for context size 1 ($F_{\beta=1}$ = 88.83). The clause end prediction part (2) worked best with the compressed chunk format while using feature combination wc (row 13) with context size 3 ($F_{\beta=1}$ = 81.61). Table 2 shows a monotonic increase of the F rates for increasing context size. Indeed, the increase goes on for context size 4 but its maximal F rate (81.72) is probably not significantly higher than the one for context size 3. We have combined the two results with the heuristic rules to a complete clause structure which obtained $F_{\beta=1}$ = 71.34 on part 3 of the shared task (training data only)[2].

After finding the best training configurations for the training data, we have applied these to the development and the test data for the shared task. The results can be found in table 3. All F rates are better than the baseline scores (Tjong Kim Sang and Déjean, 2001). Recall scores are lower than precision scores, like we have observed in our earlier work. Predicting clause ends seems to be more difficult than predicting clause starts. We do not know what could be causing this.

[2]For part 3 of the task we have also attempted to predict clause ends while using clause start information. This generated improved clause end results (83.50) but the complete clause results did not change much (71.39).

[3]After removing a software bug, the numbers in table 3 marked by * differ from the paper proceedings version.

development	precision	recall	$F_{\beta=1}$	
part 1	92.94%	86.87%	89.80	*[3]
part 2	83.80%	80.44%	82.09	
part 3	76.54%	67.20%	71.57	*

test	precision	recall	$F_{\beta=1}$	
part 1	92.91%	85.08%	88.82	*
part 2	84.72%	79.96%	82.28	
part 3	76.91%	60.61%	67.79	*

Table 3: Results obtained for the development and the test data set for the three parts of the shared task.

4 Concluding Remarks

We have put forward a method for identifying clauses in sentences given the words, their part-of-speech tags and a base chunk structure of the sentence: the CoNLL-2001 shared task. It uses a memory-based learner for predicting positions of clause starts and clause ends. After this, a list of heuristic rules is used for converting these positions to a consistent embedded clause structure. Our approach obtains $F_{\beta=1}$ = 66.67 on the test data of the third part of the shared task.

References

Michael Collins. 1999. *Head-Driven Statistical Models for Natural Language Processing.* PhD thesis, University of Pennsylvania.

Walter Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2000. *TiMBL: Tilburg Memory Based Learner, version 3.0, Reference Guide.* ILK Technical Report 00-01. http://ilk.kub.nl/.

David M. Magerman. 1995. Statistical Decision-Tree Models for Parsing. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (ACL'95).* Cambridge, MA, USA.

Erik F. Tjong Kim Sang and Hervé Déjean. 2001. Introduction to the CoNLL-2001 Shared Task: Clause Identification. In *Proceedings of the CoNLL-2001.* Toulouse, France.

Erik F. Tjong Kim Sang and Jorn Veenstra. 1999. Representing Text Chunks. In *Proceedings of EACL'99.* Bergen, Norway.

Erik F. Tjong Kim Sang. 2000. Text Chunking by System Combination. In *Proceedings of CoNLL-2000 and LLL-2000.* Lisbon, Portugal.