Tagging the Dutch PAROLE Corpus

Jesse de Does, John van der Voort van der Kleij

Instituut voor Nederlandse Lexicologie

This article was published in: Mariët Theune et al. (eds.), *Computational Linguistics in the Netherlands 2001, Selected Papers from the Twelfth CLIN Meeting*, Rodopi, Amsterdam - New York, 2002, p. 62-76.

Abstract

We discuss the annotation with part of speech and lemma of the Dutch PAROLE Internet Corpus.

The PAROLE PoS tagger is a combination of statistical taggers. It includes the Markov tagger TnT and 3 taggers developed at the INL[1] with the purpose of using other information besides the training data. Lemma is assigned by a deterministic procedure, based on an extensive lexicon.

The output is in some respects not entirely satisfactory; we discuss what can be done about this without having to manually correct the complete corpus.

1 Introduction

In 2002, our department intends to make a retrieval application for the Dutch PAROLE corpus available on-line[2]. The application will enable the user, among other things, to query the corpus for occurrences of word types, lemma and part of speech, and combinations, both boolean and proximity-based, of these criteria.

This paper describes the annotation with part of speech and lemma. The corpus and tagset (section 2) were developed in the two European PAROLE projects (cf. http://www.inl.nl/eng/corp/parole.htm). For lemmatising, we use the PAROLEX lexicon, which is described in section 3. Sentence splitting is performed by means of a deterministic program (section 4). For the part of speech tagging proper, we investigated various statistical taggers, which are briefly described in section 5, and compared them on the PAROLE corpus (section 6). The final PAROLE tagger is a combination of statistical taggers (section 7). After the statistical tagging phase, some post-processing is performed, which is described in sections 8 and 9. We give conclusions in section 10, and indicate future directions in section 11.

2 PAROLE corpus and tagset

The complete corpus consists of about 20 million tokens. The composition is mixed: about 65% is newspaper text (NRC and Meppeler Courant), 16% books, 12% news (NOS journaal, jeugdjournaal), 7% periodicals (Waterkampioen, Waddenbulletin, Liberaal Reveil).

[1]One of which uses TiMBL (Daelemans, Zavrel, Van der Sloot and Van den Bosch 2001).
[2]Cf. http://www.inl.nl/corp/corp.htm.

The rather detailed tagset (211 tags) is a language-specific subset of the multilingual PAROLE tagset, which is based on the EAGLES guidelines (Dutilh, Raaijmakers and Kruyt 1996, Kruyt 1998, Dutilh and Kruyt 2002).

For an overview of the parts of speech and features cf. figure 1. Since no syntactic annotation is planned for the corpus, emphasis in the tag assignment method is on context-dependent (‘functional’) rather than formal distinctions (Dutilh and Kruyt 2002)[3]. Some features are rather difficult for automatic taggers, like the distinction between transitive (function=trans) and intransitive (function=int) verbal use or the distinction between main verb (type=main) and copula (type=cop)/auxiliary (type=aux).

NOU(n)	type ∈ {*common, proper*}, gender ∈ {*fm, n, -*}, number ∈ {*sg, pl, -*}
ADJ(ective)	type=*quali(tative)*, degree ∈ {*pos(itive), comp(arative), sup(erlative)*}, infl(ection) ∈ {*basic, inflected*}
ADV(erb)	type ∈ {*general, prononimal*}, degree ∈ {*pos(itive), comp(arative), sup(erlative),-*}
ADP(osition)	type=*pre(position)*
NUM(eral)	type ∈ {*cardinal, ordinal, -*}, number ∈ {*sg, pl, -*}
V(e)RB	type ∈ {*main, aux(iliary), cop(ula)*}, mood ∈ {*ind(icative), inf(initive), imper(ative), conj(unctive), part(iciple)*}, tense ∈ {*pres(ent), impf, past, -*}, person ∈ {*1, 2, 3, -*}, number ∈ {*sg, pl, -*}, function ∈ {*trans(itive), intrans(itive), impers(onal), refl(exive), -*}
PR(o)N(oun)	type ∈ {*pers(onal), dem(onstrative), rel(ative), excl(amative), ind(e)f(inite), inter(rogative), rec(i)p(rocal), refl(exive)*}, person ∈ {*1, 2, 3, -*}, gender ∈ {*f, m, n, -*}, number ∈ {*sg, pl, -*}, case ∈ {*nom, dat, acc, -*}
DET(erminer)	type ∈ {*dem, excl, inter, poss, rel, indf*}, person ∈ {*1, 2, 3, -*}, gender ∈ {*f, m, n, o (multilabel), -*}, number ∈ {*sg, pl, -*}
ART(icle)	type ∈ {*def, indf*} gender ∈ {*n, o (multilabel), -*}, number ∈ {*sg, pl, -*}
CON(junction)	type ∈ {*coor(dinating), sub(ordinating)*}
INT(erjection)	–
UNIQUE	type=*inf(initive)mark(er)*
RES(idual)	type ∈ {*acro(nym), abbr(eviation), trunc(ation), -*}

Table 1: The Parole corpus tagset, parts of speech and features

The training corpus, part of which was developed in the LE-PAROLE project, has 15% book, 60% news, 25% citations from newspapers and periodicals. It consists of 100,000 tokens; in view of the fine-grained tagset and the mixed composition of the corpus, this is rather small, as might be seen, for instance, from the fact that the tag *VRB(main,ind,impf,1,sg,trs)* (first person singular of transitive main verb, past tense) appears only once in the training corpus.

Summarizing, an accurate tagging of the complete PAROLE corpus may be a difficult task because of the mixed composition of the corpus, the amount of detail in the tagset, and the lack of training data.

[3]E.g. an adverbially used adjective is tagged ‘adverb’.

3 The lexicon

We have been working on the lexicon since 1992, when a first version was used by our tagger-lemmatiser *DutchTale*. It consisted of types with part of speech and lemma information. After the publication of the *Woordenlijst Nederlandse taal* (*Woordenlijst Nederlandse Taal* 1995), the types and lemmas were adapted to the new spelling (Van der Voort van der Kleij and Kruyt 1997). Types in the pre-1995 spelling were not deleted, but got their lemma respelled. About 30,000 new lemmas with their frequently attested word-forms were added, and about 14,000 outdated lemmas and their word-forms were deleted. This tagging lexicon also had proper names.

In the LE-PAROLE project, the participants developed a special tagset and a computational PAROLE lexicon (Kruyt 1998). For 20,000 frequent types, the many features of this tagset were added to the part-of-speech information. In the PAROLE lexicon, almost all Dutch function words (which, taken together, account for about 50% of the tokens in a text) are present. For the remaining ca. 200,000 types of the DutchTale lexicon, the PAROLE features have been added during the past few years. The adapted tagging lexicon is called PAROLEX.

We finally note that the lexicon is corpus-based: almost all types have been attested with reasonable frequency in texts.

4 Sentence splitting

Sentence boundaries in both the training data and the PAROLE corpus are marked by means of a rule-based program. The program has been developed in Perl at INL.

The input is ‘raw text’ i.e. text without html or sgml text markup. The program bases its decisions on evaluation of the left and right context of potential sentence boundaries (occurrences of a dot, question or exclamation mark). The decision criteria are based on a careful pattern analysis of a corpus of journal material of more than 1.5 million tokens. A similar pattern-matching has been explored by Grefenstette (Grefenstette 1999).

5 Statistical taggers investigated

This section is about the individual taggers we considered for inclusion in the PAROLE combination tagger (section 7.2) which we use for the tagging of the complete corpus. For a similar comparison of algorithms see (De Pauw and Daelemans 2000).

5.1 Statistical tagging: algorithms

The statistical taggers investigated in this paper use four different algorithms: Markov trigram tagging, Memory-based learning, AdaBoost classification and Maximum Entropy classification. Below, we briefly explain these algorithms.

- *Markov taggers* maximize a criterion of the form $\Pi_{t=1,T} P(T_t|\mathbf{h})P(w_t|T_t)$ over a sentence. Here the history **h** consists of a left window of two (trigram tagger) tags assigned to tokens preceding w_t. The probabilities are estimated on the basis of the frequencies of tag trigrams in the training corpus.

- In *memory based learning*, a new instance is classified on the bases of examples from the training data set which are closest to the new instance in some distance measure (metric). The number of examples involved is a parameter setting (referred to as $k = ...$ in the sequel). Various options are available for the metric used. The metric which gave the best results in our case is the (Modified) Value Difference Metric (MVDM) which compares to feature values V_1 and V_2 by looking at the differences in the observed conditional distributions of the target classes C_i: $\delta(V_1, V_2) = \sum_{i=1}^{n} |P(C_i|V_1) - P(C_i|V_2)|$.

- For a binary classification task, the *AdaBoost* algorithm (Schapire 1999) uses a linear decision criterion of the form $H(x) = sign \sum_{t_1}^{T} (\alpha_t h_t(x))$. Here the h_t are a set of $\{+1, -1\}$-valued functions (called *weak hypotheses* in the boosting jargon) chosen in T rounds. In each round, one of the α_i is updated, together with a set of weights D_j on the the training examples, where higher weights are assigned to misclassified examples.

- *Maximum Entropy models*. One starts with a number of (binary) *features* $f_i(C, x)$ (like, e.g. ”the target class is Noun and the type preceding the current token is *de*”). A conditional distribution of the exponential form $p(C|x) = \frac{1}{Z(x)} \sum e^{\alpha_i f_i(C,x)}$ is then computed which satisfies a number of observed *constraints* of the form $P(C_k|f_i(C_k, x) = 1) = p_i$. Among all distributions satisfying the constraints, this distribution has maximal entropy.

5.2 Use of existing taggers

In the literature on combining taggers (Van Halteren, Zavrel and Daelemans 2001), usually a few generally available ones are used: the rule-based tagger RBT (Brill 1995)[4], the memory-based tagger MBT (Daelemans, Zavrel and Berck 1996), the maximum entropy tagger MXPOST (Ratnaparkhi 1996)[5] and the Markov trigram tagger TnT (Brants 2000)[6]. We tested the publicly available taggers (see section 6 for tagger results) on the training data to decide whether they should be included in the PAROLE combination tagger (cf. section 7.2).

We ended up including only TnT, which had the best performance among the freely available taggers on our data (cf. section 6).

We have not tested MBT, since we were already using TiMBL with classification features very similar to MBT (cf. section 5.3). MBT is based on TiMBL, so

[4]The tagger can be obtained at http://www.cs.jhu.edu/˜brill/RBT1_14.tar.Z.
[5]Tagger at http://www.cis.upenn.edu/˜adwait/statnlp.html.
[6]Tagger at http://www.coli.uni-sb.de/˜thorsten/tnt/.

we expect MBT to add little new information to the combination. Besides, using TiMBL directly gives use more freedom to experiment with different algorithm parameter settings. We had the RBT tagger running on the training data for two weeks without terminating[7]. The program does not appear to be suitable for large-sized tagsets; cf. the attempts to train it on the WOTAN/2 tagset (Van Halteren et al. 2001). MXPOST was not included because its performance on our corpus is rather disappointing: on the full tagset it did not even achieve the accuracy of our best unigram model.

5.3 Taggers developed at INL

As we wanted to try to use other sources of information besides the training data, we chose to develop a few taggers of our own. There is nothing particularly new or exciting about the design of these taggers. The aim was merely to integrate some information into the taggers which is not available to the freely available ones.

Our 3 taggers are all based on a general-purpose classification program. We used TiMBL (Daelemans et al. 2001)[8] to implement memory-based classification, and simple C^{++} implementations of AdaBoost and conditional Maximum Entropy modeling to construct an AdaBoost and a Maximum Entropy tagger. We shall refer to these taggers in the sequel as ‘Memory based’, ‘AdaBoost’ and ‘Maximum Entropy’ taggers.

Choosing classification features

The features used in our taggers are for a large part similar to the ones used in other statistical taggers: words and tags in the context of the type to tag, suffix letters, capitalization. Two classes of features were added.

We use “ambitags” based on our lexicon. An *ambitag* is a symbol representing the set of tags a type may assume in a lexicon or corpus, for instance *VRB(mood=part,tense=past)*|*ADJ(infl=basic)* (past participle or adjective). Ambitags are also used in MBT, where they are taken from the training data[9]. We follow Brants (2000) in assigning distinct ambitags to uppercase and lowercase tokens.

We also try to extract information from unannotated data by using a distributional clustering. Distributional clustering is an unsupervised technique which derives a set of word classes automatically from an untagged corpus. Word types occurring in similar contexts are clustered together. There is no a-priori reason why the classes thus obtained should be related to the tags in our tagset.

Here, we use mutual information clustering (Brown, Della Pietra, deSouza and Mercer 1990), with 100 word classes, of the types with frequency ≥ 10 in the 20

[7]We thank an anonymous reviewer for pointing to a more efficient implementation at http://nlp.cs.jhu.edu/cgi-bin/cgiwrap/˜rflorian.
[8]The package can be obtained from http://ilk.kub.nl.
[9]MBT’s ambitags depend on the relative frequency of the different taggings of the type in the training corpus.

million word corpus. The cluster ID of the type being tagged is used as a classification feature. Both TiMBL and the AdaBoost algorithm consider this feature quite informative.

5.4 Mode of execution

There are three distinct ways of actually using a classification algorithm to tag text:

1. *Simple*: Independent classification of each separate token. This does not imply no context at all is involved, it means that tags assigned by the tagger to neighbouring tokens cannot be used as classification features. Instead, we can (and do) use other properties of neighbouring tokens (like the "ambitags") as context features.

2. *Feedback*: Use assigned tags of tokens preceding the current token as classification feature. This mode is used by MBT.

3. *Viterbi*: Both tags preceding and following the current token are used. One tags a sentence with the sequence of tags that maximizes the joint probability of the tag-token sequence. This is done by Markov taggers. Also MXPOST uses dynamic programming.

The taggers used in our combination tagger are currently of the simple type. We have not tried out Viterbi mode yet; taking past experience into account (Abney, Schapire and Singer 1999, De Pauw and Daelemans 2000), we should perhaps not expect too much of it. Using feedback did lead to some improvement in the maximum entropy tagger (cf. table 2), but this effect was no longer noticeable in the combination tagger.

Further details of implementation

We implemented maximum entropy classification by means of the 'Improved Iterative Scaling' parameter estimation algorithm.

Memory based classification was tried out with different parameter settings. The TiMBL package allows choosing the number of neighbours and the metric. We obtained the best results by choosing 9 neighbours and using the Modified Value Difference Metric.

The details of the implementation of the AdaBoost tagger may need some elucidation. We opted for the Multiple Classifier implementation of AdaBoost (AdaBoost/MI in Abney et al. 1999). For each target class, a separate classifier is trained; the class which gets the highest confidence score in the associated class membership decision problem is chosen. There is no reason why the confidence scores of different classifiers should be directly comparable, so this procedure is not very well founded theoretically. It has obtained reasonable results in practice, cf. Nakagawa, Kudoh and Matsumoto 2001[10], Abney et al. 1999.

[10]Using Support Vector Machine classification.

The training time of our implementation increases linearly with both the size of the tagset and the size of the training data. We therefore trained the tagger in a hierarchical way, so it would decide on a coarser tag before proceeding to assign fine-grained distinctions. As a by-product of this approach, the performance on the coarser level improves slightly.

Hierarchical classification has been applied to maximum entropy models by Goodman (Goodman 2000).

6 Comparison of individual taggers

6.1 Testing procedure

We tested the individual taggers on a fixed split of the training corpus in 10,000 tokens of test data and 90,000 tokens of training data and by performing 10-fold crossvalidation[11] on the training corpus.

To illustrate how much the context contributes to the tagging accuracy, we give some scores for unigram models. In addition to the baseline performance, we also include the results of unigram taggers which only use formal and lexical features of the current token.

Evaluation on different levels

It would not be very illuminating to include results for the full tagset only, since it is not directly comparable to other efforts, so we specify accuracy on three levels: part of speech (13 tags), light tagset (82 tags)[12], full tagset (211 tags).

6.2 Results

Table 2 gives the results for the fixed split, table 3 gives the results of 10-fold crossvalidation on the last 90,000 tokens of the trainingcorpus. The lower score in crossvalidation is caused by diversity in text type. Tagging a linguistic text (Van Sterkenburg 1984) is not easy for a tagger mainly trained on news data, cf. table 4.

Significance

Using McNemar's chi-square test (Dietterich 1998), we find that most tagger score comparisons in the tables are significant at a confidence threshold of 0.05. Somewhat unreliable are the scores given for PoS on the small test set of 10,000 tokens. Here, for instance, whereas the comparison *TnT* $<$ *AdaBoost* is significant, the comparisons *TnT* $<$ *Memory based*, or *Memory based* $<$ *AdaBoost* are not significant according to the test.

[11]This means 10 different complementary splits of the training data in a test set and a training set are used.

[12]The light tagset omits some difficult features like main verb function and pronoun case.

Tagger	full tagset	light tagset	PoS
unigram (TnT, no unk. word model)	80.75	86.34	91.60
unigram (TnT, with unk. word model)	86.20	90.53	93.01
unigram (Maximum Entropy, lexicon, no clusters)	87.63	91.86	94.07
unigram (Maximum Entropy, lexicon, clusters)	88.06	92.14	94.27
MXPOST	86.76	92.53	94.91
Memory based, k=9	**90.78**	94.75	96.83
Maximum Entropy (feedback)	**90.80**	95.01	96.89
TnT	89.83	94.39	96.53
Memory based, MVDM metric, k=1	90.31	94.58	96.75
Maximum Entropy (simple)	90.51	94.79	96.86
AdaBoost	90.64	**95.17**	**97.12**

Table 2: Performance of individual taggers, fixed split of training corpus

Tagger	full tagset	light tagset	PoS
TnT	87.97	92.30	94.87
Memory-based ($k = 9$)	**89.48**	92.74	95.30
AdaBoost	89.03	**93.10**	**95.62**

Table 3: Performance of individual taggers, evaluated by crossvalidation on the training corpus

Tagger	full tagset	light tagset	PoS
TnT	85.59	88.96	92.22
Memory based ($k = 9$)	86.07	88.52	91.94
AdaBoost	86.41	89.29	92.72

Table 4: Performance of individual taggers – tagging a linguistic text while trained on news text

Conclusions

- Performance on the full tagset is still unsatisfactory.
- Integrating other data (besides the training corpus) into the taggers leads to a certain increase in accuracy.
- The hierarchical tagging procedure implemented in the AdaBoost tagger does lead to a slightly better performance on the coarse-grained level.

6.3 Analysis of tagger errors

This section is based mainly on the output of the AdaBoost tagger. We chose not to use the PAROLE combination tagger (section 7.2) for this purpose; crossvalidation would have taken too much training time.

Part of Speech

The confusion matrix of part of speech (table 5) is based on the output of the AdaBoost tagger. Here, 'RES(idual)' appears to be the most difficult part of speech for the tagger (and other taggers as well) to recognize. The subcategorization with *type=–* of the RES tag is used in the PAROLE tagset for foreign words and unclassifiable items, such as garbled words. The errors are mostly due to the difficult distinction between RES(type=-) and NOU (noun). It is hardly surprising that ADV(erb) and ADJ(ective) are difficult to distinguish. The confusion between VRB (verb) and ADJ(ective) is caused by (past) participles, which are tagged as adjectives when used adnominally.

	ADJ	ADP	ADV	ART	CON	DET	INT	NOU	NUM	PRN	RES	UNIQUE	VRB
ADJ	5560	4	276	0	0	14	0	185	1	2	12	0	167
ADP	1	12069	23	0	67	0	0	8	0	0	9	11	1
ADV	257	36	7661	1	88	32	2	100	0	40	4	8	36
ART	0	0	0	11020	0	0	0	2	3	23	2	0	0
CON	0	67	68	0	4376	7	0	1	0	73	3	0	2
DET	13	0	46	1	8	2104	0	6	0	93	0	0	42
INT	1	0	0	0	1	0	46	18	0	1	1	0	1
NOU	117	1	56	3	0	5	4	23448	52	14	116	0	179
NUM	13	0	0	36	0	0	0	30	2207	0	31	0	2
PRN	3	0	30	84	92	96	1	13	1	3847	2	0	1
RES	30	21	12	37	22	1	0	634	67	7	1569	2	17
UNIQUE	0	5	4	0	0	0	0	0	0	0	0	774	0
VRB	79	1	22	0	1	18	1	149	4	3	2	0	12464

Table 5: Part of Speech

Features

Table 6 gives confusion matrices for some of the most difficult features.

As could be expected, distinguishing transitive and intransitive verb use is difficult for a part of speech tagger. We did not expect reflexive use would be so difficult to detect (precision 56%, recall 45%). The category *imp(ersonal)* (*het regent=it is raining*) is not understood by the tagger at all (precision 33%, recall 25%). Distinguishing verb ‘type’ (this feature distinguishes main verb, auxiliary and copula), which is also a context-dependent category in our system, is almost equally difficult. Another problematic item in verb subcategorisation is the distinction between infinitive and present plural, as can be seen in the confusion table for VRB.person[13].

function	-	imp	int	ref	trs
-	4386	31	289	15	198
imp	39	27	35	0	5
int	304	22	1846	12	476
ref	3	0	12	87	93
trs	215	2	368	42	3950

type	aux	cop	mai
aux	2636	372	386
cop	271	1107	147
mai	411	150	6977

person	-	1	2	3
-	3750	9	5	287
1	23	140	1	48
2	5	1	120	30
3	303	22	19	7693

Table 6: Difficult verbal features

	dem	excl	indf	inter	pers	recp	refl	rel
dem	484	0	0	0	1	0	0	38
excl	0	0	0	1	0	0	0	2
indf	0	0	469	1	85	0	1	11
inter	1	2	13	1	2	0	0	15
pers	5	0	88	0	1563	0	5	0
recp	0	0	0	0	0	46	0	0
refl	0	0	0	0	9	0	262	0
rel	28	0	19	7	1	0	0	683

Table 7: PRN.type, accuracy 91.29%

[13]Infinitives are marked with person=–.

7 Combination of taggers

7.1 Serial connection

A relatively unexplored way of exploiting multiple classifiers is serial connection: tagger T_2 tries to learn from the mistakes made by tagger T_1[14]. To be more precise, we give tagger T_2 information about T_1's tag assignment to the current token and its immediate neighbours.

In the case where taggers T_1 and T_2 use the same tagset and training data, we found the serial combination not to be better than the best individual taggers.

However, there is a way of applying this scheme which does give a slight gain in accuracy. When T_1 is trained on a different (i.c. larger) dataset (with a different tagset), we can use T_2 as a translator between the two tagsets. In our case, T_1 is TnT trained on the VU-version of the Eindhoven Corpus, and T_2 is an AdaBoost tagger, which uses a window of 5 Eindhoven tags and 5 neighbouring words as context model, together with a standard unknown word model as discussed previously.

Here are the results for the serial combination[15]:

Method	full tagset	light	PoS
TnT	89.83	94.39	96.53
Memory based	90.78	94.75	96.83
AdaBoost	90.64	95.17	97.12
TnT Eindhoven + AdaBoost	91.14	95.60	97.43

7.2 Parallel connection

More usual is parallel connection of classifiers by means of diverse *voting* schemes. Voting in its most elementary form means several classifiers are run, and the target class which obtains most 'votes' (is chosen by most classifiers) is assigned. The scheme can be sophisticated in various ways, for instance by weighing the votes of the individual classifiers according to the observed accuracy of the classifiers.

The ingredients for the combination we now use to tag the complete corpus are TnT, TnT-Eindhoven (TnT trained on the VU-version of the Eindhoven corpus, which is tagged with the original Uit den Boogaart tagset (Uit den Boogaart 1975)), the AdaBoost tagger, the maximum entropy tagger, and the Memory based tagger with one neighbour[16].

Since we want to use the information contained in the larger Eindhoven tagged corpus, the combination method must be a form of 'stacking'. *Stacking* is combining classifiers by applying a classification algorithm the features of which are the tags chosen by the participant classifiers. Voting presupposes identical tagsets for the participating taggers. The tagging of the Spoken Dutch Corpus (Oostdijk 2000)

[14]Our interest in this scheme was rearoused by a discussion with prof. W. Daelemans.

[15]The combination taggers were only tested on a fixed split of the training corpus.

[16]$k = 9$ Is more accurate, but slower.

also combined taggers with different tagsets (Zavrel and Daelemans 2000).

We tried several combination methods for the PAROLE combination tagger, and finally settled on the TagPair method of (Van Halteren et al. 2001), which is somewhere halfway between a voting method and a stacked classifier and performs well when not too much training data is present.

Tagger	full tagset	light	PoS
TnT	89.83	94.39	96.53
Memory based	90.78	94.75	96.83
AdaBoost	90.64	95.17	97.12
TnT + TnT-Eindhoven + Memory-based + Max.Ent. + AdaBoost	92.03	95.99	97.63

To conclude, whereas the combination improves the PoS and 'light' tagging to a quite decent level, the accuracy measured on the full tagset remains somewhat disappointing.

7.3 Evaluation

As stated, we aimed to compensate for the small training dataset. Did we succeed in doing so?

In this section, we first compare taggers trained on the PAROLE training corpus with TnT trained on the Eindhoven Corpus (which has approximately 750,000 tokens). We measure performance on distinctions present in both tagsets. Both tagsets distinguish a plural common noun from a plural indicative verb form.

Tagger	NOU precision	NOU recall	VRB precision	VRB recall
TnT-Eindhoven	0.962	0.952	0.955	0.964
AdaBoost	0.948	0.951	0.952	0.949
Maximum Entropy	0.932	0.940	0.940	0.932
TnT	0.940	0.940	0.941	0.948
Memory based	0.931	0.927	0.928	0.932

On this essential distinction the tagger trained on the Eindhoven corpus outperforms all individual taggers trained on the PAROLE training corpus.

Next, we compare TnT-Eindhoven to the combination tagger of section 7.2 using the fixed split of the training data.

Tagger	NOU precision	NOU recall	VRB precision	VRB recall
TnT-Eindhoven	0.947	0.947	0.930	0.930
TnT + TnT-Eindhoven + Memory-based + Max.Ent. + AdaBoost	0.951	0.987	0.981	0.927

The combination appears to do slightly better (although the amount of test data is such that this is hardly significant). So we may have succeeded in compensating for the lack of training data.

8 Lemmatising and type-based correction

Presently, no lemma is derived for types not found in the lexicon; no morphological analysis of unknown words is performed.

The PAROLEX lexicon lists a set $S(w)$ of pairs of PoS tag and lemma for each type w; we assign as lemma those lemmata from $S(w)$ which agree with the tag assignment by the tagger.

Lexical information is also used to improve the output of the tagger. After an analysis of the complete tagged corpus, we extended our lexicon.

We now assume that the following classes are completely covered in the lexicon: articles, adpositions, determiners, pronouns and conjunctions. We completely checked lists of adverbs, abbreviations and acronyms produced by the tagger. Numerals, which cannot be completely listed, were checked by means of a simple formal pattern. This manual correction has not been completed yet.

Thus, the lexical component is able to block certain tag assignments. The information that a certain type can *not* have a certain part of speech is fed back into the voting process, which then produces the best-scoring tag which is *not* dissaproved by the lexical check.

It is difficult to quantify the error reduction achieved by this process. It obviously does not contribute to disambiguating truly ambiguous types, which are responsible for the larger part of the tagger errors. So the improvement of the tagging accuracy on a per-token basis is probably not very significant.

However, one must bear in mind that the corpus retrieval application presents a different view of the data. Many users just search for context for certain types. Having searched for a type, they are first presented with a list of its assigned taggings. The output from the statistical tagging process, if viewed in this way, sometimes looks downright absurd. For example, at some point all parts of speech are assigned to the type *ons* by tagger; the acceptable assignments are in the vast majority, but one simply cannot let the absurd ones survive in the corpus. It would appear that the type-based correction is able to remove many of the most embarrassing errors statistical taggers tend to produce.

9 Rule-based corrections

As we have seen (table 6), our statistical taggers are rather bad at distinguishing main verbs from auxiliaries and infinitives from plural present forms. Using regular expression patterns formulated in terms of PoS tags, we can correct some of the errors.

Thus, we are for instance able to improve the tagger accuracy on VRB.type from 86.06% to 89.8% (when postprocessing the output of the AdaBoost tagger). There is an obvious drawback to this approach: it assumes all tags in the relevant context to be correct, except for the one investigated. Indeed, this drawback manifests itself clearly when we apply the same rule to the output of the Memory based tagger: there is less improvement here (from 88.95% to 90.93%).

10 Conclusions

We are on the whole a bit disappointed by the achievied tagging accuracy. We invested quite a lot of time trying out various statistical techniques, but the returns for each separate effort were rather small. Most of our attempts did indeed yield a certain improvement, but it was often not significant enough.

We do not expect to be able to improve contextual disambiguation significantly without increasing the training corpus.

The remaining errors are still partly of a non-contextual nature. We do expect to be able to reduce these.

11 Future directions

In the short run, we will extend the rule-based component to improve the PAROLE tagging.

In the long run, our experiences will be relevant for a new project at our institute, the Integrated Language Database of 8th-21st-Century Dutch (Kruyt 2000). Apart from dictionary and lexicon data, the ILD will contain a text corpus covering all these centuries. The corpus will be annotated with lemma and PoS. We have stressed the importance of the amount of training data for a reliable PoS tagging. In designing a tagset to cover Dutch of varying age and regional origin, we intend to reuse existing resources. This means an effort will be made to harmonize this tagset with existing Dutch tagsets. Among the datasets and tagsets that need to be considered are the Eindhoven Corpus (with various taggings), the Spoken Dutch Corpus, and the Early Middle Dutch Corpus developed at the INL, a corpus of about 1.5 million tokens, annotated with fine-grained PoS tags and lemma.

References

Abney, S., Schapire, R. and Singer, Y.(1999), Boosting applied to tagging and PP attachment, *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Brants, T.(2000), TnT – a statistical part-of-speech tagger, *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, Seattle, WA.

Brill, E.(1995), Transformation-Based Error-Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging, *Computational Linguistics* **21**(4), 543–565.

Brown, P. F., Della Pietra, V. J., deSouza, P. V. and Mercer, R. L.(1990), Class-based n-gram models of natural language, *Proceedings of the IBM Natural Language ITL*, Paris, France, pp. 283–298.

Daelemans, W., Zavrel, J. and Berck, S.(1996), MBT: A MemoryBased Part of Speech Tagger-Generator.

Daelemans, W., Zavrel, J., Van der Sloot, K. and Van den Bosch, A.(2001), TiMBL: Tilburg Memory Based Learner, version 4.0, Reference Guide.

ILK Technical Report 01-04, *Technical report*, ILK, available from http://ilk.kub.nl/downloads/pub/papers/ilk0104.ps.gz.

De Pauw, G. and Daelemans, W.(2000), The Role of Algorithm Bias vs Information Source in Learning Algorithms for Morphosyntactic Disambiguation, *Proceedings of the Fourth Conference on Computational Language Learning (CoNLL-2000)*, Lissabon, Portugal, pp. 19–24.

Dietterich, T. G.(1998), Approximate statistical test for comparing supervised classification learning algorithms, *Neural Computation* **10**(7), 1895–1923.

Dutilh, T. and Kruyt, T.(2002), Implementation and Evaluation of PAROLE PoS in a National Context, *Proceedings of the third international conference on language resources and evaluation (LREC-2002)*, Las Palmas, pp. 1615–1621, also at http://www.inl.nl/eng/pub/publist/tbpub.htm.

Dutilh, T., Raaijmakers, S. and Kruyt, T.(1996), Tagset for Dutch Morphosyntactic Corpus Annotation. Parole Task 4.1.4a. INL Working Papers 96-02, *Technical report*.

Goodman, J.(2000), Classes for Fast Maximum Entropy Training.

Grefenstette, G.(1999), Tokenization, *in* H. van Halteren (ed.), *Syntactic Word Class Tagging*, Kluwer.

Kruyt, J. G.(1998), Elektronische woordenboeken en tekstcorpora voor Europese taaltechnologie, *Trefwoord* **12**, 28–42, also at http://www.inl.nl/pub/tref.htm.

Kruyt, J. G.(2000), Towards the Integrated Language Database of 8th-21st Century Dutch, *Revue française de linguistique appliquée* **V-2**, 33–44, also at http://www.inl.nl/taalbank/.

Nakagawa, T., Kudoh, T. and Matsumoto, Y.(2001), Unknown word guessing and part-of-speech tagging using support vector machine, *Proceedings of the Sixth Natural Language Processing Pacific Rim Symposium*, Tokyo, Japan, pp. 325–331.

Oostdijk, N.(2000), Het corpus gesproken nederlands, *Nederlandse Taalkunde* **5**, 280–284.

Ratnaparkhi, A.(1996), A Maximum Entropy Model for Part-of-Speech Tagging, *in* E. Brill and K. Church (eds), *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, Somerset, New Jersey, pp. 133–142.

Schapire, R. E.(1999), A Brief Introduction to Boosting, *IJCAI*, pp. 1401–1406.

Uit den Boogaart, P. C.(1975), *Woordfrequenties*, Oosthoek, Scheltema & Holkema, Utrecht.

Van der Voort van der Kleij, J. and Kruyt, J. G.(1997), Lexicon for a linguistic annotation of Dutch text, *TELRI newsletter* (5), 32–35.

Van Halteren, H., Zavrel, J. and Daelemans, W.(2001), Improving accuracy in word class tagging through combination of machine learning systems., *Computational Linguistics* **27**(2), 199–230.

Van Sterkenburg, P.(1984), *Van woordenlijst tot woordenboek. Inleiding tot de geschiedenis van woordenboeken van het Nederlands*, E.J. Brill, Leiden.

Woordenlijst Nederlandse Taal(1995), SDU, Den Haag.

Zavrel, J. and Daelemans, W.(2000), Bootstrapping a Tagged Corpus through Combination of Existing Heterogeneous Taggers, *Proceedings of the second international conference on language resources and evaluation (LREC-2000)*, Athens, Greece.