

A Comparison of Two Different Approaches to Morphological Analysis of Dutch

Guy De Pauw¹, Tom Laureys², Walter Daelemans¹, Hugo Van hamme²

¹ University of Antwerp
CNTS - Language Technology Group
Universiteitsplein 1
2610 Antwerpen (Belgium)
firstname.lastname@ua.ac.be

² K.U.Leuven
ESAT
Kasteelpark Arenberg 10
3001 Leuven (Belgium)
firstname.lastname@esat.kuleuven.ac.be

Abstract

This paper compares two systems for computational morphological analysis of Dutch. Both systems have been independently designed as separate modules in the context of the FLaVoR project, which aims to develop a modular architecture for automatic speech recognition. The systems are trained and tested on the same Dutch morphological database (CELEX), and can thus be objectively compared as morphological analyzers in their own right.

1 Introduction

For many NLP and speech processing tasks, an extensive and rich lexical database is essential. Even a simple word list can often constitute an invaluable information source. One of the most challenging problems with lexicons is the issue of out-of-vocabulary words. Especially for languages that have a richer morphology such as German and Dutch, it is often unfeasible to build a lexicon that covers a sufficient number of items. We can however go a long way into resolving this issue by accounting for novel productions through the use of a limited lexicon and a morphological system.

This paper describes two systems for morphological analysis of Dutch. They are conceived as part of a morpho-syntactic language model for inclusion in a modular speech recognition engine being developed in the context of the FLaVoR project (Demuynek et al., 2003). The FLaVoR project investigates the feasibility of using powerful linguistic information in the recognition process. It is generally acknowledged that more accurate linguistic knowledge sources improve on speech recognition accuracy, but are only rarely incorporated into the recognition process (Rosenfeld, 2000). This is due to the fact that the architecture of most current speech recognition systems requires all knowledge sources to be compiled into the recognition process at run

time, making it virtually impossible to include extensive language models into the process.

The FLaVoR project tries to overcome this restriction by using a more flexible architecture in which the search engine is split into two layers: an acoustic-phonemic decoding layer and a word decoding layer. The reduction in data flow performed by the first layer allows for more complex linguistic information in the word decoding layer. Both morpho-phonological and morpho-syntactic modules function in the word decoding process. Here we focus on the morpho-syntactic model which, apart from assigning a probability to word strings, provides (scored) morphological analyses of word candidates. This morphological analysis can help overcome the previously mentioned problem of out-of-vocabulary words, as well as enhance the granularity of the speech recognizer's language model.

Successful experiments on introducing morphology into a speech recognition system have recently been reported for the morphologically rich languages of Finnish (Siivola et al., 2003) and Hungarian (Szarvas and Furui, 2003), so that significant advances can be expected for FLaVoR's target language Dutch as well. But as the modular nature of the FLaVoR architecture requires the modules to function as stand-alone systems, we are also able to evaluate and compare the modules more generally as morphological analyzers in their own right, which can be used in a wide range of natural language applications such as information retrieval or spell checking.

In this paper, we describe and evaluate these two independently developed systems for morphological analysis: one system uses a machine learning approach for morphological analysis, while the other system employs finite state techniques. After looking at some of the issues when dealing with Dutch morphology in section 2, we discuss the architecture of the machine learn-

ing approach in section 3, followed by the finite state method in section 4. We discuss and compare the results in section 5, after which we draw conclusions.

2 Dutch Morphology: Issues and Resources

Dutch can be situated between English and German if we define a scale of morphological richness in Germanic languages. It lacks certain aspects of the rich inflectional system of German, but features a more extensive inflection, conjugation and derivation system than English. Contrary to English, Dutch for instance includes a set of diminutive suffixes: e.g. *appel+tje* (*little apple*) and has a larger set of suffixes to handle conjugation.

Compounding in Dutch can occur in different ways: words can simply be concatenated (e.g. *plaats+bewijs* (*seat ticket*)), they can be conjoined using the ‘s’ infix (e.g. *toegang+s+bewijs* (*entrance ticket*)) or the ‘e(n)’ infix (e.g. *fles+en+mand* (*bottle basket*)). In Dutch affixes are used to produce derivations: e.g. *aanvaard+ing* (*accept-ance*).

Morphological processes in Dutch account for a wide range of spelling alternations. For instance: a syllable containing a long vowel is written with two vowels in a closed syllable (e.g. *poot* (*paw*)) or with one vowel in an open syllable (e.g. *poten* (*paws*)). Consonants in the coda of a syllable can become voiced (e.g. *huis - huizen* (*house(s)*) or doubled (e.g. *kip - kippen* (*chicken(s)*)). These and other types of spelling alternations make morphological segmentation of Dutch word forms a challenging task. It is therefore not surprising to find that only a handful of research efforts have been attempted. (Heemskerk, 1993; Dehaspe et al., 1995; Van den Bosch et al., 1996; Van den Bosch and Daelemans, 1999; Laureys et al., 2002). This limited number may to some extent also be due to the limited amount of Dutch morphological resources available.

The Morphological Database of CELEX

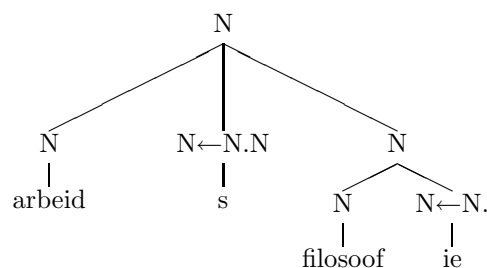
Currently, CELEX is the only extensive and publicly available morphological database for Dutch (Baayen et al., 1995). Unfortunately, this database is not readily applicable as an information source in a practical system due to both a considerable amount of annotation errors and a number of practical considerations. Since both of the systems described in this pa-

per are data-driven in nature, we decided to semi-automatically make some adjustments to allow for more streamlined processing. A full overview of these adjustments can be found in (Laureys et al., 2004) but we point out some of the major problems that were rectified:

- Annotation of diminutive suffix and unanalyzed plurals and participles was added (e.g. *appel+tje*).
- Inconsistent treatment of several suffixes was resolved (e.g. *acrobataat+isch* (*acrobat+ic*) vs. *agnostisch* (*agnostic*)).
- Truncation operations were removed (e.g. *filosoof+(isch+)* (*philosophy*)).

The Task: Morphological Segmentation

The morphological database of CELEX contains hierarchically structured and fully tagged morphological analyses, such as the following analysis for the word ‘arbeidsfilosofie’ (*labor philosophy*):



The systems described in this paper deal with the most important subtask of morphological analysis: segmentation, i.e. breaking up a word into its respective morphemes. This type of analysis typically also requires the modeling of the previously mentioned spelling changes, exemplified in the above example (*arbeidsfilosofie* → *arbeid+s+filosoof+ie*). In the next 2 sections, we will describe two different approaches to the segmentation/alternation task: one using a machine-learning method, the other using finite state techniques. Both systems however were trained and tested on the same data, i.e. the Dutch morphological database of CELEX.

3 A Machine Learning Approach

One of the most notable research efforts modeling Dutch morphology can be found in Van den Bosch and Daelemans (1999). Van den Bosch and Daelemans (1999) define morphological analysis as a classification task that can be learned from labeled data. This is accomplished

at the level of the grapheme by recording a local context of five letters before and after the focus letter and associating this context with a morphological classification which not only predicts a segmentation decision, but also a graphemic (alternation) and hierarchical mapping.

The system described in Van den Bosch and Daelemans (1999) employs the IB1-IG memory-based learning algorithm, which uses information-gain to attribute weighting to the features. Using this method, the system is able to achieve an accuracy of 64.6% of correctly analyzed word forms. On the segmentation task alone, the system achieves a 90.7% accuracy of correctly segmented words. On the morpheme level, a 94.5% F-score is observed.

Towards a Cascaded Alternative

The machine learning approach to morphological analysis described in this paper is inspired by the method outlined in Van den Bosch and Daelemans (1999), but with some notable differences. The first difference is the data set used: rather than using the extended morphological database, we concentrated on the database excluding inflections and conjugated forms. These morphological processes are to a great extent regular in Dutch. As derivation and compounding pose the most challenging task when modeling Dutch morphology, we have opted to concentrate on those processes first. This allows us to evaluate the systems with a clearer insight into the quality of the morphological analyzers with respect to the hardest tasks.

Further, the systems described in this paper use the adjusted version of CELEX described in section 2, instead of the original dataset. The main reason for this can be situated in the context of the FLaVoR project: since our morphological analyzer needs to operate within a speech recognition engine, it is paramount that our analyzers do not have to deal with truncated forms, as it would require us to hypothesize unrealized forms in the acoustic input stream. Even though using the modified dataset does not affect the general applicability of the morphological analyzer itself, it does entail that a direct comparison with the results in Van den Bosch and Daelemans (1999) is not possible.

The overall design of our memory-based system for morphological analysis differs from the one described in Van den Bosch and Daelemans (1999) as our approach takes a more traditional stance with respect to classification. Rather

than encoding different types of classification in conglomerate classes, we have set up a cascaded approach in which each classification task (spelling alternation, segmentation) is handled separately. This allows us to identify problems at each point in the task and enables us to optimize each classification step accordingly. To avoid percolation of bad classification decisions at one point throughout the entire classification cascade, we ensure that all solutions are retained throughout the entire process, effectively turning later classification steps into re-ranking mechanisms.

Alternation

The input of the morphological analyzer is a word form such as ‘arbeidsfilosofie’. As a first step to arrive at the desired segmented output ‘arbeid+s+filosoof+ie’, we need to account for the spelling alternation. This is done as a precursor to the segmentation task, since preliminary experiments showed that segmentation on a word form like ‘arbeidsfilosoofie’ is easier to model accurately than segmentation on the initial word form.

First, we record all possible alternations on the training set. These range from general alternations like doubling the vowel of the last syllable (e.g. arbeidsfilosoof) to very detailed, almost word-specific alternations (e.g. Europa → euro). Next, these alternations in the training set are annotated and an instance base is extracted. Table 1 shows an example of instances for the word ‘aanbidder’ (*admirer*). In this example we see that alternation number 3 is associated with the double ‘d’, denoting a deletion of that particular letter.

	Precision	Recall	F-score
MBL	80.37%	88.12%	84.07%

Table 2: Results for alternation experiments

These instances were used to train the memory-based learner TiMBL (Daelemans et al., 2003). Table 2 displays the results for the alternation task on the test set. Even though these appear quite modest, the only restriction we face with respect to consecutive processing steps lies in the recall value. The results show that 255 out of 2,146 alternations in the test set were not retrieved. This means that we will not be able to correctly analyze 2.27% of the test set (which contains 11,256 items).

Left Context					Focus	Right Context					Combined			Class
-	-	-	-	-	a	a	n	b	i	d	-a	-aa	aan	0
-	-	-	-	a	a	n	b	i	d	d	-aa	aan	anb	0
-	-	-	a	a	n	b	i	d	d	e	aan	anb	nbi	0
-	-	a	a	n	b	i	d	d	e	r	anb	nbi	bid	0
-	a	a	n	b	i	d	d	e	r	-	nbi	bid	idd	0
a	a	n	b	i	d	d	e	r	-	-	bid	idd	dde	0
a	n	b	i	d	d	e	r	-	-	-	idd	dde	der	3
n	b	i	d	d	e	r	-	-	-	-	dde	der	er-	0
b	i	d	d	e	r	-	-	-	-	-	der	er-	r-	0

Table 1: Alternation instances for ‘aanbidder’

Segmentation

A memory-based learner trained on an instance base extracted from the training set constitutes the segmentation system. An initial feature set was extracted using a windowing approach similar to the one described in Van den Bosch and Daelemans (1999). Preliminary experiments were however unable to replicate the high segmentation accuracy of said method, so that extra features needed to be added. Table 3 shows an example of instances extracted for the word ‘rijksontvanger’ (*state collector*). Experiments on a held-out validation set confirmed both left and right context sizes determined in Van den Bosch and Daelemans (1999)¹. The last two features are combined features from the left and right context and were shown to be beneficial on the validation set. They denote a group containing the focus letter and the two consecutive letters and a group containing the focus letter and the three previous letters respectively.

A numerical feature (‘Dist’ in Table 3) was added that expresses the distance to the previous morpheme boundary. This numerical feature avoids overeager segmentation, i.e. a small value for the distance feature has to be compensated by other strong indicators for a morpheme boundary. We also consider the morpheme that was compiled since the last morpheme boundary (features in the column ‘Current Morpheme’). A binary feature indicates whether or not this morpheme can be found in the lexicon extracted from the training set. The next two features consider the morpheme formed by adding the next letter in line.

Note however that the introduction of these features makes it impossible to precompile the instance base for the test set, since for instance

the distance to the previous morpheme boundary can obviously not be known before actual segmentation takes place. We therefore set up a server application and generated instances on the fly.

1,141,588 instances were extracted from the training set and were used to power a TiMBL server. The optimal algorithmic parameters were determined with cross-validation on the training set². A client application extracted instances from the test set and sent them to the server on the fly, using the previous output of the server to determine the value of the above-mentioned features. We also adjusted the verbosity level of the output so that confidence scores were added to the classifier’s decision.

A post-processing step generated all possible segmentations for all possible alternations. The possible segmentations for the word ‘apotheker’ (*pharmacist*) for example constituted the following set: {(apothek)(er), (apotheker), (apotheker), (apothek)(er)}. Next, the confidence scores of the classifier’s output were multiplied for each possible segmentation to express the overall confidence score for the morpheme sequence. Also, a lexicon extracted from the training set with associated probabilities was used to compute the overall probability of the morpheme sequence (using a Laplace-type smoothing process to account for unseen morphemes). Finally, a bigram model computed the probability of the possible morpheme sequences as well.

Table 4 describes the results at different stages of processing and expresses the number of words that were correctly segmented. Only using the confidence scores output by the memory-based learner (equivalent to using a non-ranking

¹Context size was restricted to four graphemes for reasons of space in Table 3.

²IB1-IG was used with Jeffrey divergence as distance metric, no weighting, considering 11 nearest neighbors using inverse linear weighting.

Left Context				Focus	Right Context				Dist	Current Morpheme		Next Morpheme		Combined		Class
-	-	-	-	r	i	j	k	s	0	r	1	ri	0	rij	-r	0
-	-	-	r	i	j	k	s	o	1	ri	0	rij	1	ijk	-ri	0
-	-	r	i	j	k	s	o	n	2	rij	1	rijk	1	jks	-rij	0
-	r	i	j	k	s	o	n	t	3	rijk	1	rijks	0	kso	rijk	1
r	i	j	k	s	o	n	t	v	0	s	1	so	0	son	ijks	1
i	j	k	s	o	n	t	v	a	0	o	0	on	1	ont	jkso	0
j	k	s	o	n	t	v	a	n	1	on	1	ont	1	ntv	kson	0
k	s	o	n	t	v	a	n	g	2	ont	1	ontv	0	tva	sont	0
s	o	n	t	v	a	n	g	e	3	ontv	0	ontva	0	van	ontv	0
o	n	t	v	a	n	g	e	r	4	ontva	0	ontvan	0	ang	ntva	0
n	t	v	a	n	g	e	r	-	5	ontvan	0	ontvang	1	nge	tvan	0
t	v	a	n	g	e	r	-	-	6	ontvang	1	ontvange	0	ger	vang	1
v	a	n	g	e	r	-	-	-	0	e	1	er	1	er-	ange	0
a	n	g	e	r	-	-	-	-	1	er	1	er-	0	r-	nger	1

Table 3: Instances for Segmentation Task for the word ‘rijksontvanger’.

Ranking Method	Full Word Score
MBL	81.36%
Lexical	84.56%
Bigram	82.44%
MBL+Lexical	86.37%
MBL+Bigram	85.79%
MBL+Lexical+Bigram	87.57%

Table 4: Results at different stages of post-processing for segmentation task

approach) achieves a low score of 81.36%. Using only the lexical probabilities yields a better score, but the combination of the two achieves a satisfying 86.37% accuracy. Adding bigram probabilities to the product further improves accuracy to 87.57%. In Section 5 we will look at the results of the memory-based morphological analyzer in more detail.

4 A Finite State Approach

Since the invention of the two-level formalism by Kimmo Koskeniemi (Koskeniemi, 1983) finite state technology has been the dominant framework for computational morphological analysis. In the FLaVoR project a finite state morphological analyzer for Dutch is being developed. We have several motivations for this. First, until now no finite state implementation for Dutch morphology was freely available. In addition, finite state morphological analysis can be considered a kind of reference for the evaluation of other analysis techniques. In the current project, however, most important is the inherent bidirectionality of finite state morphologi-

cal processing. This bidirectionality should allow for a flexible integration of the morphological model in the speech recognition engine as it leaves open a double option: either the morphological system acts in *analysis* mode on word hypotheses offered by the recognizer’s search algorithm, or the system works in *generation* mode on morpheme hypotheses. Only future practical implementation of the complete recognition system will reveal which option is preferable.

After evaluation of several finite state implementations it was decided to implement the current system in the framework of the Xerox finite state tools, which are well described and allow for time and space efficient processing (Beesley and Karttunen, 2003). The output of the finite state morphological analyzer is further processed by a filter and a probabilistic score function, as will be detailed later.

Morphotactics and Orthographic Alternation

The morphological system design is a composition of finite state machines modeling morphotactics and orthographic alternations. For morphotactics a lexicon of 29,890 items was generated from the training set (118 prefixes, 189 suffixes, 3 infixes and 29,581 roots). The items were divided in 23 lexicon classes, each of which could function as an item’s continuation class. The resulting finite state network has 24,858 states and 61,275 arcs.

The Xerox finite state tools allow for a specification of orthographical alternation by means of (conditional) replace rules. Each replace

rule compiles into a single finite state transducer. These transducers can be put in cascade or in parallel. In the case at hand, all transducers were put in cascade. The resulting finite state transducer contains 3,360 states and 81,523 arcs. The final transducer (a composition of the lexical network and the orthographical transducer) contains 29,234 states and 106,105 arcs.

Dealing with Overgeneration

As the finite state machine has no memory stack³, the use of continuation classes only allows for rather crude morphotactic modeling. For example, in ‘on-ont-vlam-baar’ (*un-inflame-able*) the noun ‘vlam’ first combines with the prefix ‘ont’ to form a verb. Next, the suffix ‘baar’ is attached and an adjective is built. Finally, the prefix ‘on’ negates the adjective. This example shows that continuation classes cannot be strictly defined: the suffix ‘baar’ combines with a verb but immediately follows a noun root, while the prefix ‘on’ requires an adjective but is immediately followed by another prefix. Obviously, such a model leads to overgeneration. In practice, the average number of analyses per test set item is 7.65. The maximum number of analyses is 1,890 for the word ‘belastingsadministratie’ (*tax administration*).

In section 3 the numerical feature ‘Dist’ was used to avoid overeager segmentation. We apply a similar kind of filter to the segmentations generated by the morphological finite state transducer. A penalty function for short morphemes is defined: 1- and 2-character morphemes receive penalty 3, 3-character morphemes penalty 1. Both an absolute and relative⁴ penalty threshold are set. Optimal threshold values (11 and 2.5 respectively) were determined on the basis of the training set. Analyses exceeding one of both thresholds are removed. This filter proves quite effective as it reduces the average number of analyses per item with 36.6% to 4.85.

Finally, all remaining segmentation hypotheses are scored and ranked using an N-gram morpheme model. We applied a bigram and trigram model, both using Katz back-off and modified Kneser-Ney smoothing. The bigram slightly

³Actually, the Xerox finite state tools do allow for a limited amount of ‘memory’ by a restricted set of unification operations termed *flag diacritics*. Yet, they are insufficient for modeling long distance dependencies with hierarchical structure.

⁴Relative to the number of morphemes.

outperformed the trigram model, showing that the training data is rather sparse. Tables 5, 6 and 7 all show results obtained with the bigram model.

Monomorphemic Items

The biggest remaining problem at this stage of development is the scoring of monomorphemic test items which are not included as word roots in the lexical finite state network. Sometimes these items do not receive any analysis at all, in which case we correctly consider them monomorphemic. Mostly however, monomorphemes are wrongly analyzed as morphologically complex. Scoring all test items as potentially monomorphemic does not offer any solution, as the items at hand were not in the training data and thus receive just the score for unknown items. This problem of spurious analyses accounts for 57.23% of all segmentation errors made by the finite state system.

5 Comparing the Two Systems

System	1-best	2-best	3-best
Baseline	18.64		
MBM	87.57	91.20	91.68
FSM	89.08	90.87	91.01

Table 5: Full Word Scores (%) on the segmentation task

To evaluate both morphological systems, we defined a training and test set. Of the 124,136 word forms in CELEX, 110,627 constitute the training set. The test set is further split up into words with only one possible analysis (11,256 word forms) and words with multiple analyses (2,253). Since the latter set requires morphological processes beyond segmentation, we focus our evaluation on the former set in this paper. For the machine learning experiments, we also defined a held-out validation set of 5,000 word forms, which is used to perform parameter optimization and feature selection.

Tables 5, 6 and 7 show a comparison of the results⁵. Table 5 describes the percentage of words in the test set that have been segmented correctly. We defined a baseline system which considers all words in the test set as monomorphemic. Obviously this results in a very low

⁵MBM: the memory-based morphological analyzer, FSM: the finite state morphological analyzer

full word score (which shows us that 18.64% of the words in the test set are actually monomorphemic). The finite state system seems to have a slight edge over the memory-based analyzer when we looking at the single best solution. Yet, when we consider 2-best and 3-best scores, the memory-based analyzer in turn beats the finite state analyzer.

System	Precision	Recall	$F_{\beta=1}$
Baseline	18.64	07.94	11.14
MBM	91.63	90.52	91.07
FSM	89.60	94.00	91.75

Table 7: Precision and Recall Scores (%) (morphemes) on the segmentation task

We also calculated Precision and Recall on morpheme boundaries. The results are displayed in Table 6. This score expresses how many of the morpheme boundaries have been retrieved. We make a distinction between word-internal morpheme boundaries and all morpheme boundaries. The former does not include the morpheme boundaries at the end of a word, while the latter does. We provide the latter in reference to Van den Bosch and Daelemans (1999), but the focus lies on the results for word-internal boundaries as these are non-trivial. We notice that the memory-based system outperforms the finite state system, but the difference is once again small. However, when we look at Table 7 in which we calculate the amount of full morphemes that have been correctly retrieved (meaning that both the start and end-boundary have been correctly placed), we see that the finite state method has the advantage.

Slight differences in accuracy put aside, we find that both systems achieve similar scores on this dataset. When we look at the output, we do notice that these systems are indeed performing quite well. There are not many instances where the morphological analyzer cannot be said to have found a decent alternative analysis to the gold standard one. In many cases, both systems even come up with a more advanced morphological analysis: e.g. 'gekwetst' (*hurt*) is featured in the database as a monomorphemic artefact. Both systems described in this paper correctly segment the word form as 'ge+kwets+t', even though they have not yet specifically been designed to handle this type of inflection.

When performing an error analysis of the out-

put, one does notice a difference in the way the systems have tried to solve the erroneous analyses. The finite state method often seems to generate more morpheme boundaries than necessary, while the reverse is the case for the memory-based system, which seems too eager to revert to monomorphemic analyses when in doubt. This behavior might also explain the reversed situation when comparing Table 6 to Table 7. Also noteworthy is the fact that almost 60% of the errors is made on wordforms that both systems were not able to analyze correctly. Work is currently also underway to improve the performance by combining the rankings of both systems, as there is a large degree of complementarity between the two systems. Each system is able to uniquely find the correct segmentation for about 5% of the words in the test set, yielding an upperbound performance of 98.75% on the full word score for an optimally combined system.

6 Conclusion

Current work in the project focuses on further developing the morphological analyzer by trying to provide part-of-speech tags and hierarchical bracketing properties to the segmented morpheme sequences in order to comply with the type of analysis found in the morphological database of CELEX. We will further try to incorporate other machine learning algorithms like maximum entropy and support vector machines to see if it is at all possible to overcome the current accuracy threshold. Algorithmic parameter 'degradation' will be attempted to entice more greedy morpheme boundary placement in the raw output, in the hope that the post-processing mechanism will be able to properly rank the extra alternative segmentations. Finally, we will experiment on the full CELEX data set (including inflection) as featured in Van den Bosch and Daelemans (1999).

In this paper we described two data-driven systems for morphological analysis. Trained and tested on the same data set, these systems achieve a similar accuracy, but do exhibit quite different processing properties. Even though these systems were originally designed to function as language models in the context of a modular architecture for speech recognition, they constitute accurate and elegant morphological analyzers in their own right, which can be incorporated in other natural language applications as well.

System	Precision		Recall		$F_{\beta=1}$	
	All	Intern	All	Intern	All	Intern
Baseline	100	0	42.59	0	59.74	0
MBM	94.15	89.71	93.00	87.81	93.57	88.75
FSM	90.25	83.58	94.68	90.73	92.41	87.01

Table 6: Precision and Recall Scores (%) (morpheme boundaries) on the segmentation task

Acknowledgements

The research described in this paper was funded by IWT in the GBOU programme, project FLVoR: Flexible Large Vocabulary Recognition: Incorporating Linguistic Knowledge Sources Through a Modular Recogniser Architecture. (Project number 020192). <http://www.esat.kuleuven.ac.be/spch/projects/FLVoR>.

References

- R.H. Baayen, R. Piepenbrock, and L. Gulikers. 1995. *The Celex Lexical Database (Release2) [CD-ROM]*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia, U.S.A.
- K. R. Beesley and L. Karttunen, editors. 2003. *Finite State Morphology*. CSLI Publications, Stanford.
- W. Daelemans, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2003. TiMBL: Tilburg memory based learner, version 5.0, reference guide. ILK Technical Report 01-04, Tilburg University. Available from <http://ilk.kub.nl>.
- L. Dehaspe, H. Blockeel, and L. De Raedt. 1995. Induction, logic and natural language processing. In *Proceedings of the joint ELSNET/COMPULOG-NET/EAGLES Workshop on Computational Logic for Natural Language Processing*.
- K. Demuyne, T. Laureys, D. Van Compernelle, and H. Van hamme. 2003. Flavor: a flexible architecture for LVCSR. In *Proceedings of the 8th European Conference on Speech Communication and Technology*, pages 1973–1976, Geneva, Switzerland, September.
- J. Heemskerk. 1993. A probabilistic context-free grammar for disambiguation in morphological parsing. Technical Report 44, ITK, Tilburg University.
- K. Koskenniemi. 1983. *Two-level morphology: A general computational model for word-form recognition and production*. Ph.D. thesis, Department of General Linguistics, University of Helsinki.
- T. Laureys, V. Vandeghinste, and J. Duchateau. 2002. A hybrid approach to compounds in LVCSR. In *Proceedings of the 7th International Conference on Spoken Language Processing*, volume I, pages 697–700, Denver, U.S.A., September.
- T. Laureys, G. De Pauw, H. Van hamme, W. Daelemans, and D. Van Compernelle. 2004. Evaluation and adaptation of the CELEX Dutch morphological database. In *Proceedings of the 4th International Conference on Language Resources and Evaluation*, Lisbon, Portugal, May.
- R. Rosenfeld. 2000. Two decades of statistical language modeling: Where do we go from here? *Proceedings of the IEEE*, 88(8):1270–1278.
- V. Siivola, T. Hirsimäki, M. Creutz, and M. Kurimo. 2003. Unlimited vocabulary speech recognition based on morphs discovered in an unsupervised manner. In *Proceedings of the 8th European Conference on Speech Communication and Technology*, pages 2293–2296, Geneva, Switzerland, September.
- M. Szarvas and S. Furui. 2003. Finite-state transducer based modeling of morphosyntax with applications to Hungarian LVCSR. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pages 368–371, Hong Kong, China, May.
- A. Van den Bosch and W. Daelemans. 1999. Memory-based morphological analysis. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 285–292, New Brunswick, U.S.A., September.
- A. Van den Bosch, W. Daelemans, and A. Weijters. 1996. Morphological analysis as classification: an inductive-learning approach. In K. Oflazer and H. Somers, editors, *Proceedings of the Second International Conference on New Methods in Natural Language Processing, NeMLaP-2, Ankara, Turkey*, pages 79–89.