Sentence Compression for Automated Subtitling: A Hybrid Approach

Vincent Vandeghinste and **Yi Pan**
Centre for Computational Linguistics
Katholieke Universiteit Leuven
Maria Theresiastraat 21
BE-3000 Leuven
Belgium
vincent.vandeghinste@ccl.kuleuven.ac.be, yi.pan@ccl.kuleuven.ac.be

Abstract

In this paper a sentence compression tool is described. We describe how an input sentence gets analysed by using a.o. a tagger, a shallow parser and a subordinate clause detector, and how, based on this analysis, several compressed versions of this sentence are generated, each with an associated estimated probability. These probabilities were estimated from a parallel transcript/subtitle corpus. To avoid ungrammatical sentences, the tool also makes use of a number of rules. The evaluation was done on three different pronunciation speeds, averaging sentence reduction rates of 40% to 17%. The number of reasonable reductions ranges between 32.9% and 51%, depending on the average estimated pronunciation speed.

1 Introduction

A sentence compression tool has been built with the purpose of automating subtitle generation for the deaf and hard-of-hearing. Verbatim transcriptions cannot be presented as the subtitle presentation time is between 690 and 780 characters per minute, which is more or less 5.5 seconds for two lines (ITC, 1997), (Dewulf and Saerens, 2000), while the average speech rate contains a lot more than the equivalent of 780 characters per minute.

The actual amount of compression needed depends on the speed of the speaker and on the amount of time available after the sentence. In documentaries, for instance, there are often large silent intervals between two sentences, the speech is often slower and the speaker is off-screen, so the available presentation time is longer. When the speaker is off-screen, the synchrony of the subtitles with the speech is of minor importance. When subtitling the news the speech rate is often very high so the amount of reduction needed to allow the synchronous presentation of subtitles and speech is much greater. The sentence compression rate is a parameter which can be set for each sentence.

Note that the sentence compression tool described in this paper is not a subtitling tool. When subtitling, only when a sentence needs to be reduced, and the amount of reduction is known, the sentence is sent to the sentence compression tool. So the sentence compression tool is a module of an automated subtitling tool. The output of the sentence compression tool needs to be processed according to the subtitling guidelines like (Dewulf and Saerens, 2000), in order to be in the correct lay-out which makes it usable for actual subtitling. Manually post-editing the subtitles will still be required, as for some sentences no automatic compression is generated.

In real subtitling it often occurs that the sentences are not compressed, but to keep the subtitles synchronized with the speech, some sentences are entirely removed.

In section 2 we describe the processing of a sentence in the sentence compressor, from input to output. In section 3 we describe how the system was evaluated and the results of the evaluation. Section 4 contains the conclusions.

2 From Full Sentence to Compressed Sentence

The sentence compression tool is inspired by (Jing, 2001). Although her goal is text summarization and not subtitling, her sentence compression system could serve this purpose.

She uses multiple sources of knowledge on which her sentence reduction is based. She makes use of a corpus of sentences, aligned with human-written sentence reductions which is similar to the parallel corpus we use (Vandeghinste and Tjong Kim Sang, 2004). She applies a syntactic parser to analyse the syntactic structure of the input sentences. As there was no syntactic parser available for Dutch (Daelemans and Strik, 2002), we created *ShaRPA* (Vandeghinste, submitted), a shallow rule-based parser which could give us a shallow parse tree of the input sentence. Jing uses several other knowledge sources, which are not used (not available for Dutch) or not yet used in our system (like WordNet).

In figure 1 the processing flow of an input sentence is sketched.

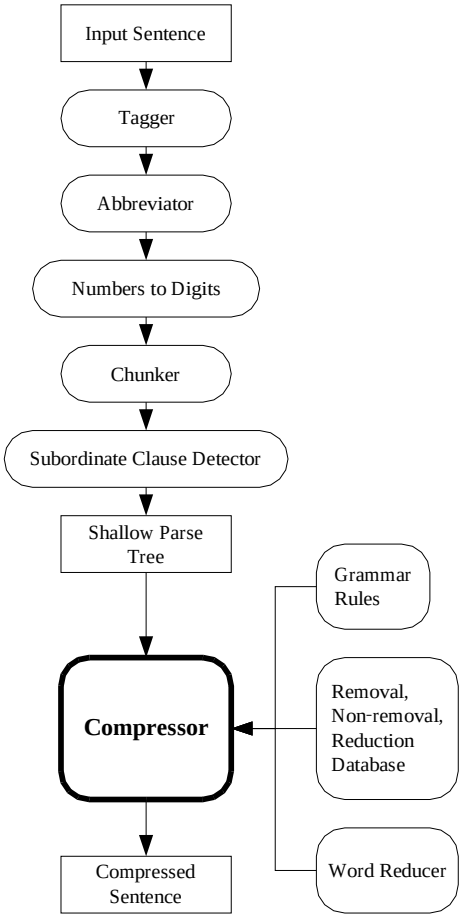


Figure 1: Sentence Processing Flow Chart

First we describe how the sentence is analysed (2.1), then we describe how the actual sentence compression is done (2.2), and after that we describe how words can be reduced for extra compression (2.3). The final part describes the selection of the ouput sentence (2.4).

2.1 Sentence Analysis

In order to apply an accurate sentence compression, we need a syntactic analysis of the input sentence.

In a first step, the input sentence gets tagged for parts-of-speech. Before that, it needs to be transformed into a valid input format for the part-of-speech tagger. The tagger we use is TnT (Brants, 2000) , a hidden Markov trigram tagger, which was trained on the Spoken Dutch Corpus (CGN), Internal Release 6. The accuracy of TnT trained on CGN is reported to be 96.2% (Oostdijk et al., 2002).

In a second step, the sentence is sent to the *Abbreviator*. This tool connects to a database of common abbreviations, which are often pronounced in full words (E.g. *European Union* becomes *EU*) and replaces the full form with its abbreviation. The database can also contain the tag of the abbreviated part (E.g. the tag for *EU* is *N(eigen,zijd,ev,basis,stan)* [E: singular non-neuter proper noun]).

In a third step, all numbers which are written in words in the input are replaced by their form in digits. This is done for all numbers which are smaller than one million, both for cardinal and ordinal numerals.

In a fourth step, the sentence is sent to ShaRPa, which will result in a shallow parse-tree of the sentence. The chunking accuracy for noun phrases (NPs) has an F-value of 94.7%, while the chunking accuracy of prepositional phrases (PPs) has an F-value of 95.1% (Vandeghinste, submitted).

A last step before the actual sentence compression consists of rule-based clause-detection: Relative phrases (RELP), subordinate clauses (SSUB) and OTI-phrases (OTI is *om ... te + infinitive*[1]) are detected. The accuracy of these detections was evaluated on 30 files from the CGN component of read-aloud books, which contained 7880 words. The evaluation results are presented in table 1.

Type of S	Precision	Recall	F-value
OTI	71.43%	65.22%	68.18%
RELP	69.66%	68.89%	69.27%
SSUB	56.83%	60.77%	58.74%

Table 1: Clause Detection Accuracy

The errors are mainly due to a wrong analysis of coordinating conjunctions, which is not only the weak point in the clause-detection module, but also in ShaRPa. A full parse is needed to accurately solve this problem.

2.2 Sentence Compression

For each chunk or clause detected in the previous steps, the probabilities of removal, non-removal and reduction are estimated. This is described in more detail in 2.2.1.

Besides the statistical component in the compression, there are also a number of rules in the compression program, which are described in more detail in 2.2.2.

The way the statistical component and the rule-based component are combined is described in 2.2.3.

[1]There is no equivalent construction in English. OTI is a VP-selecting complementizer.

2.2.1 Use of Statistics

Chunk and clause removal, non-removal and reduction probabilities are estimated from the frequencies of removal, non-removal and reduction of certain types of chunks and clauses in the parallel corpus.

The parallel corpus consists of transcripts of television programs on the one hand and the subtitles of these television programs on the other hand. A detailed description of how the parallel corpus was collected, and how the sentences and chunks were aligned is given in (Vandeghinste and Tjong Kim Sang, 2004).

All sentences in the source corpus (transcripts) and the target corpus (subtitles) are analysed in the same way as described in section 2.1, and are chunk aligned. The chunk alignment accuracy is about 95% (F-value).

We estimated the removal, non-removal and reduction probabilities for the chunks of the types NP, PP, adjectival phrase (AP), SSUB, RELP, and OTI, based on their chunk removal, non-removal and reduction frequencies.

For the tokens not belonging to either of these types, the removal and non-removal probabilities were estimated based on the part-of-speech tag for those words. A reduced tagset was used, as the original CGN-tagset (Van Eynde, 2004) was too fine-grained and would lead to a multiplication of the number of rules which are now used in ShaRPa. The first step in SharPa consists of this reduction.

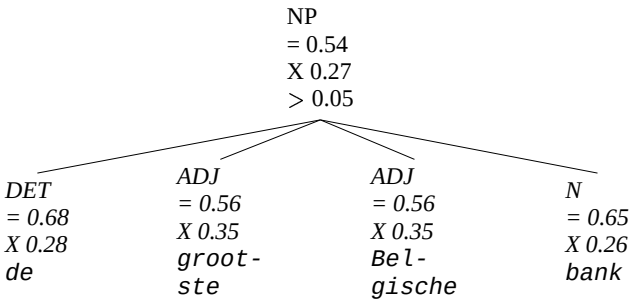
For the PPs, the SSUBs and the RELPs, as well as for the adverbs, the chunk/tag information was considered as not fine-grained enough, so the estimation of the removal, non-removal and reduction probabilities for these types are based on the first word of those phrases/clauses and the reduction, removal and non-removal probabilities of such phrases in the parallel corpus, as the first words of these chunk-types are almost always the heads of the chunk. This allows for instance to make the distinction between several adverbs in one sentence, so they do not all have the same removal and non-removal probabilities. A disadvantage is that this approach leads to sparse data concerning the less frequent adverbs, for which a default value (average over all adverbs) will be employed.

An example : A noun phrase.

`de grootste Belgische bank`

[E: the largest Belgian bank]

After tagging and chunking the sentence and after detecting subordinate clauses, for every non-terminal node in the shallow parse tree we retrieve the measure of removal (X), of non-removal (=) and of reduction[2] ($>$). For the terminal nodes, only the measures of removal and of non-removal are used.



For every combination the probability estimate is calculated. So if we generate all possible compressions (including no compression), the phrase `de grootste Belgische bank` will get the probability estimate $0.54 \times (0.68 \times 0.56 \times 0.56 \times 0.65) = 0.07485$. For the phrase `de Belgische bank` the probability estimate is $0.05 \times (0.68 \times 0.35 \times 0.56 \times 0.65) = 0.00433$, and so on for the other alternatives.

In this way, the probability estimate of all possible alternatives is calculated.

2.2.2 Use of Rules

As the statistical information allows the generation of ungrammatical sentences, a number of rules were added to avoid generating such sentences. The procedure keeps the necessary tokens for each kind of node. The rules were built in a bootstrapping manner

In some of these rules, this procedure is applied recursively. These are the rules implemented in our system:

- If a node is of type SSUB or RELP, keep the first word.
- If a node is of type S, SSUB or RELP, keep
  - the verbs. If there are prepositions which are particles of the verb, keep the prepositions. If there is a prepositional phrase which has a preposition which is in the complements list of the verb, keep the necessary tokens[3] of that prepositional phrase.

[2] These measures are estimated probabilities and do not need to add up to 1, because in the parallel training corpus, sometimes a match was detected with a chunk which was not a reduction of the source chunk or which was not identical to the source chunk: the chunk could be paraphrased, or even have become longer.

[3] Recursive use of the rules

  - each token which is in the list of negative words. These words are kept to avoid altering the meaning of the sentence by dropping words which negate the meaning.
  - the necessary tokens of the *te + infinitives (TI)*.
  - the conjunctions.
  - the necessary tokens of each NP.
  - the numerals.
  - the adverbially used adjectives.
- If a node is of type NP, keep
  - each noun.
  - each nominalised adjectival phrase.
  - each token which is in the list of negative words.
  - the determiners.
  - the numerals.
  - the indefinite prenominal pronouns.
- If a node is of type PP, keep
  - the preposition.
  - the determiners.
  - the necessary tokens of the NPs.
- If the node is of type adjectival phrase, keep
  - the head of the adjectival phrase.
  - the prenominal numerals.
  - each word which is in the list of negative words.
- If the node is of type OTI, keep
  - the verbs.
  - the *te + infinitives*.
- If the node is of type TI, keep the node.
- If the node is a *time phrase*[4], keep it.

These rules are chosen because in tests on earlier versions of the system, using a different test set, ungrammatical output was generated. By using these rules the output should be grammatical, provided that the input sentence was analysed correctly.

[4]A time phrase, as defined in ShaRPa is used for special phrases, like dates, times, etc. E.g. *27 september 1998, kwart voor drie* [E: quarter to three].

2.2.3 Combining Statistics and Rules

In the current version of the system, in a first stage all variations on a sentence are generated in the statistical part, and they are ranked according to their probability. In a second stage, all ungrammatical sentences are (or should be) filtered out, so the only sentence alternatives which remain should be grammatical ones.

This is true, only if tagging as well as chunking were correct. If errors are made on these levels, the generation of an ungrammatical alternative is still possible.

For efficiency reasons, a future version of the system should combine the rules and statistics in one stage, so that the statistical module only generates grammatically valid sentence alternatives, although there is no effect on correctness, as the resulting sentence alternatives would be the same if statistics and rules were better integrated.

2.3 Word Reduction

After the generation of several grammatical reductions, which are ordered according to their probability estimated by the product of the removal, non-removal and reduction probabilities of all its chunks, for every word in every compressed alternative of the sentence it is checked whether the word can be reduced.

The words are sent to a *WordSplitter*-module, which takes a word as its input and checks if it is a compound by trying to split it up in two parts: the modifier and the head. This is done by lexicon lookup of both parts. If this is possible, it is checked whether the modifier and the head can be recompounded according to the word formation rules for Dutch (Booij and van Santen, 1995), (Haeseryn et al., 1997). This is done by sending the modifier and the head to a *WordBuilding*-module, which is described in more detail in (Vandeghinste, 2002). This is a hybrid module combining the compounding rules with statistical information about the frequency of compounds with the samen head, the frequency of compounds with the same modifier, and the number of different compounds with the same head.

Only if this module allows the recomposition of the modifier and the head, the word can be considered to be a compound, and it can potentially be reduced to its head, removing the modifier.

If the words occur in a database which contains a list of compounds which should not be split up, the word cannot be reduced. For example, the word *voetbal* [E: football] can be split up and recompounded according to the word formation rules

for Dutch (*voet* [E: foot] and *bal* [E: ball]), but we should not replace the word *voetbal* with the word *bal* if we want an accurate compression, with the same meaning as the original sentence, as this would alter the meaning of the sentence too much. The word *voetbal* has (at least) two different meanings: soccer and the ball with which soccer is played. Reducing it to *bal* would only keep the second meaning. The word *gevangenisstraf* [E: prison sentence] can be split up and recompounded (*gevangenis* [E: prison] and *straf* [E: punishment]). We can replace the word *gevangenisstraf* by the word *straf*. This would still alter the meaning of the sentence, but not to the same amount as it would have been altered in the case of the word *voetbal*.

2.4 Selection of the Compressed Sentence

Applying all the steps described in the previous sections results in an ordered list of sentence alternatives, which are supposedly grammatically correct.

When word reduction was possible, the word-reduced alternative is inserted in this list, just after its full-words equivalent.

The first sentence in this list with a length smaller than the maximal length (depending on the available presentation time) is selected.

In a future version of the system, the word reduction information can be integrated in a better way with the rest of the module, by combining the probability of reduction/non-reduction of a word with the probability of the sentence alternative. The reduction probability of a word would then play its role in the estimated probability of the compressed sentence alternative containing this reduced word.

3 Evaluation

The evaluation of a sentence compression module is not an easy task. The output of the system needs to be judged manually for its accuracy. This is a very time consuming task. Unlike (Jing, 2001), we do not compare the system results with the human sentence reductions. Jing reports a succes rate of 81.3% for her program, but this measure is calculated as the percentage of decisions on which the system agrees with the decisions taken by the human summarizer. This means that 81.3% of all system decisions are correct, but does not say anything about how many sentences are correctly reduced.

In our evaluation we do not expect the compressor to simulate human summarizer behaviour. The results presented here are calculated on the sentence level: the amount of valid reduced sentences, being those reductions which are judged by human *raters* to be accurate reductions: grammatical sentences with (more or less) the same meaning as the input sentence, taking into account the meaning of the previous sentences on the same topic.

3.1 Method

To estimate the available number of characters in a subtitle, it is necessary to estimate the average pronunciation time of the input sentence, provided that it is unknown. We estimate sentence duration by counting the number of syllables in a sentence and multiplying this with the average duration per syllable (ASD).

The ASD for Dutch is reported to be about 177 ms (Koopmans-van Beinum and van Donzel, 1996), which is the syllable speed without including pauses between words or sentences.

We did some similar research on CGN using the ASD as a unit of analysis, while we consider both the situation without pauses and the situation with pauses. Results of this research are presented in table 2.

ASD	no pauses	pauses included
All files	186	237
One speaker	*185*	239
Read-aloud	188	*256*

Table 2: Average Syllable Duration (ms)

We extract the word duration from all the files in each component of CGN. A description of the components can be found in (Oostdijk et al., 2002).

We created a syllable counter for Dutch words, which we evaluated on all words in the CGN lexicon. For 98.3% of all words in the lexicon, syllables are counted correctly. Most errors occur in very low frequency words or in foreign words.

By combining word duration information and the number of syllables we can calculate the average speaking speed.

We evaluated sentence compression in three different conditions:

The fastest ASD in our ASD-research was 185 ms (one speaker, no pauses), which was used for *Condition A*. We consider this ASD as the maximum speed for Dutch.

The slowest ASD (256 ms) was used for *Condition C*. We consider this ASD to be the minimum speed for Dutch.

We created a testset of 100 sentences mainly focused on news broadcasts in which we use the real pronunciation time of each sentence in the testset which results in an ASD of 192ms. This ASD was

used for *Condition B*, and is considered as the real speed for news broadcasts.

We created a testset of 300 sentences, of which 200 were taken from transcripts of television news, and 100 were taken from the 'broadcast news' component of CGN.

To evaluate the compressor, we estimate the duration of each sentence, by counting the number of syllables and multiplying that number with the ASD for that condition. This leads to an estimated pronunciation time. This is converted to the number of characters, which is available for the subtitle.

We know the average time for subtitle presentation at the VRT (Flemish Broadcasting Coorporation) is 70 characters in 6 seconds, which gives us an average of 11.67 characters per second.

So, for example, if we have a test sentence of 15 syllables, this gives us an estimated pronunciation time of 2.775 seconds (15 syllables $\times$ 185 ms/syllable) in condition A. When converting this to the available characters, we multiply 2.775 seconds by 11.67 characters/second, resulting in 32 (2.775s $\times$ 11.67 ch/s = 32.4 ch) available characters.

In condition B (considered to be real-time) for the part of the test-sentences coming from CGN, the pronunciation time was not estimated, as it was available in CGN.

3.2 Results

The results of our experiments on the sentence compression module are presented in table 3.

Condition	A	B	C
No output (0)	44.33%	41.67%	15.67%
Avg Syllable speed (msec/syllable)	185	192	256
Avg Reduction Rate	39.93%	37.65%	16.93%
Interrater Agreement	86.2%	86.9%	91.7%
Accurate Compr.	4.8%	8.0%	28.9%
+/- Acc. Compr.	28.1%	26.3%	22.1%
Reasonable Compr.	32.9%	34.3%	51%

Table 3: Sentence Compression Evaluation on the Sentence Level

The sentence compressor does not generate output for all test sentences in all conditions: In those cases where no output was generated, the sentence compressor was not able to generate a sentence alternative which was shorter than the maximum number of characters available for that sentence. The cases where no output is generated are not considered as errors because it is often impossible, even for humans, to reduce a sentence by about 40%, without changing the content too much. The amount of test sentences where no output was generated is presented in table 3. The high percentage of sentences where no output was generated in conditions A and B is most probably due to the fact that the compression rates in these conditions are higher than they would be in a real life application. Condition C seems to be closer to the real life compression rate needed in subtitling.

Each condition has an average reduction rate over the 300 test sentences. This reduction rate is based on the available amount of characters in the subtitle and the number of characters in the source sentence.

A rater scores a compressed sentence as `+` when it is grammatically correct and semantically equivalent to the input sentence. No essential information should be missing. A sentence is scored as `+/-` when it is grammatically correct, but some information is missing, but is clear from the context in which the sentence occurs. All other compressed sentences get scored as `-`.

Each sentence is evaluated by two *raters*. The lowest score of the two raters is the score which the sentence gets. Interrater agreement is calculated on a 2 point score: if both raters score a sentence as `+` or `+/-` or both raters score a sentence as `-`, it is considered an agreed judgement. Interrater agreement results are presented in table 3.

Sentence compression results are presented in table 3. We consider both the `+` and `+/-` results as reasonable compressions.

The resulting percentages of reasonable compressions seem to be rather low, but one should keep in mind that these results are based on the sentence level. One little mistake in one sentence can lead to an inaccurate compression, although the major part of the decisions taken in the compression process can still be correct. This makes it very hard to compare our results to the results presented by Jing (2001), but we presented our results on sentence evaluations as it gives a clearer idea on how well the system would actually perform in a real life application.

As we do not try to immitate human subtitling behaviour, but try to develop an equivalent approach, our system is not evaluated in the same way as the system deviced by Jing.

4 Conclusion

We have described a hybrid approach to sentence compression which seems to work in general. The combination of using statistics and filtering out invalid results because they are ungrammatical by using a set of rules is a feasible way for automated

sentence compression.

The way of combining the probability-estimates of chunk removal to get a ranking in the generated sentence alternatives is working reasonably well, but could be improved by using more fine-grained chunk types for data collection.

A full syntactic analysis of the input sentence would lead to better results, as the current sentence analysis tools have one very weak point: the handling of coordinating conjunction, which leads to chunking errors, both in the input sentence as in the processing of the used parallel corpus. This leads to misestimations of the compression probabilities and creates noise in the behaviour of our system.

Making use of semantics would most probably lead to better results, but a semantic lexicon and semantic analysis tools are not available for Dutch, and creating them would be out of the scope of the current project.

In future research we will check the effects of improved word-reduction modules, as word reductions often seem to lead to inaccurate compressions. Leaving out the word-reduction module would probably lead to an even bigger amount of *no output*-cases. This will also be checked in future research.

5 Acknowledgements

Research funded by IWT (Institute for Innovation in Science and Technology) in the STWW program, project ATraNoS (Automatic Transcription and Normalisation of Speech). For more information visit http://atranos.esat.kuleuven.ac.be/.

We would like to thank Ineke Schuurman for rating the reduced sentences.

References

G. Booij and A. van Santen. 1995. *Morfologie. De woordstructuur van het Nederlands.* Amsterdam University Press, Amsterdam, Netherlands.

T. Brants. 2000. TnT - A Statistical Part-of-Speech Tagger. Published online at http://www.coli.uni-sb.de/thorsten/tnt.

W. Daelemans and H. Strik. 2002. Het Nederlands in Taal- en Spraaktechnologie: Prioriteiten voor Basisvoorzieningen. Technical report, Nederlandse Taalunie.

B. Dewulf and G. Saerens. 2000. Stijlboek Teletekst Ondertiteling. Technical report, VRT, Brussel. Internal Subtitling Guidelines.

W. Haeseryn, G. Geerts, J de Rooij, and M. van den Toorn. 1997. *Algemene Nederlandse Spraakkunst*. Martinus Nijhoff Uitgevers, Groningen.

ITC. 1997. Guidance on standards for subtitling. Technical report, ITC. Online at http://www.itc.org.uk/codes_guidelines/broadcasting/tv/sub_sign_audio/subtitling_stnds/.

H. Jing. 2001. *Cut-and-Paste Text Summarization*. Ph.D. thesis, Columbia University.

F.J. Koopmans-van Beinum and M.E. van Donzel. 1996. Relationship Between Discourse Structure and Dynamic Speech Rate. In *Proceedings ICSLP 1996*, Philadelphia, USA.

N. Oostdijk, W. Goedertier, F. Van Eynde, L. Boves, J.P. Marters, M. Moortgat, and H. Baayen. 2002. Experiences from the Spoken Dutch Corpus. In *Proceedings of LREC 2002*, volume I, pages 340–347, Paris. ELRA.

F. Van Eynde. 2004. Part-of-speech Tagging en Lemmatisering. Internal manual of Corpus Gesproken Nederlands, published online at http://www.ccl.kuleuven.ac.be/Papers/POSmanual_febr2004.pdf.

V. Vandeghinste and E. Tjong Kim Sang. 2004. Using a parallel transcript/subtitle corpus for sentence compression. In *Proceedings of LREC 2004*, Paris. ELRA.

V. Vandeghinste. 2002. Lexicon optimization: Maximizing lexical coverage in speech recognition through automated compounding. In *Proceedings of LREC 2002*, volume IV, pages 1270–1276, Paris. ELRA.

V. Vandeghinste. submitted. ShaRPa: Shallow Rule-based Parsing, focused on Dutch. In *Proceedings of CLIN 2003*.