Proceedings of DiSS'03, Disfluency in Spontaneous Speech Workshop, 5–8 September 2003, Göteborg University, Sweden. Robert Eklund (ed.), Gothenburg Papers in Theoretical Linguistics 90, ISSN 0349–1021, pp. 63–66.

Memory-based disfluency chunking

Piroska Lendvai†, Antal van den Bosch† & Emiel Krahmer‡

† ILK Research Group, Faculty of Arts, Tilburg University, The Netherlands
‡ Communication & Cognition, Faculty of Arts, Tilburg University, The Netherlands

Abstract

We investigate the feasibility of machine learning in automatic detection of disfluencies in a large syntactically annotated corpus of spontaneous spoken Dutch. We define disfluencies as chunks that do not fit under the syntactic tree of a sentence (including fragmented words, laughter, self-corrections, repetitions, abandoned constituents, hesitations and filled pauses). We use a memory-based learning algorithm for detecting disfluent chunks, on the basis of a relatively small set of low-level features, keeping track of the local context of the focus word and of potential overlaps between words in this context. We use *attenuation* to deal with sparse data and show that this leads to a slight improvement of the results and more efficient experiments. We perform a search for the optimal settings of the learning algorithm, which yields an accuracy of 97% and an F-score of 80%. This is a significant improvement of the baselines and of the results obtained with the default settings of the learner.

1. Introduction

Disfluencies are a main stumbling block for automatic processing of spoken language. Hence a preprocessing module capable of automatically filtering out all kinds of disfluencies would be very useful to have, because it is likely to improve further processing such as parsing and interpretation.

Various researchers have worked on automatic disfluency detection in the past two decades, including, but not limited to, Hindle [8], Bear et al. [1], Nakatani & Hirschberg [11], Heeman & Allen [7], Oviatt [13], Shriberg et al. [16]. Most of this work is largely empirical and involves relatively small datasets, since annotating corpora for disfluencies is a difficult and time-consuming process. In addition, many of these studies tend to focus on a subset of disfluent phenomena, such as repairs or fragmented words, and are usually concerned with (American) English (exceptions include Eklund & Shriberg [6] on Swedish, Spilker et al. [17] on German, and Lendvai [10] on Dutch).

In this paper we follow a different route. We apply memory-based machine learning to automatically detect disfluencies in a large syntactically annotated corpus of spontaneous spoken Dutch. We take a broad conception of disfluency: everything that does not fit under the syntactic tree of a sentence, according to the syntactic annotators. This includes fragmented words, laughter, self-corrections, repetitions, abandoned constituents, hesitations and filled pauses. The learning task is defined as follows: given an utterance (i.e., a string of words), predict where disfluent *chunks* start and where they end. This approach may be likened to syntactic phrase chunking (e.g., Tjong Kim Sang & Buchholz [18]), where the chunker in our case marks whether a word occurs inside a disfluent chunk or outside it, rather than whether a word occurs within or outside some syntactic constituent. As input to the learning task we only use low level, readily available features. No explicit feature selection is performed in the experiments since the memory-based learner is capable of determining which features are most beneficial for the learning task. We do perform an extensive search to estimate the optimal setting of the algorithm for our task, and investigate the usefulness of special attenuation techniques (Eisner [5], van den Bosch & Buchholz [2]) to compress the data set and avoid sparse data problems.

The rest of this paper is organized as follows. In Section 2 we describe the method, starting with a brief overview of the corpus we used (2.1), and the feature representations that we derived from it (2.2). In Section 2.3 we describe the memory-based classifier. The experimental set-up is outlined in 2.4. Special attention is paid to the attenuation method (2.5) and the parameter optimization routine (2.6). The baselines are given in 2.7. In Section 3 the results are presented. We end with some concluding remarks and pointers for future research in Section 4.

2. Method

2.1. Corpus

Our experiments are based on the Spoken Dutch Corpus (Corpus Gesproken Nederlands, CGN, Release 5) that contains various kinds of discourses sampled from different regions of the Netherlands and the Flemish part of Belgium. The discourses are of various levels of spontaneity ranging from television broadcasts to telephone conversations, and the number of speakers spans from 1 (newsreading) to 7 (parliamentary sessions). For more information, see Oostdijk [12] and van der Wouden et al. [19].

For the machine learning experiments we used a representative sample of 203 full discourses from CGN, consisting of 340,545 lexical tokens in 44,939 sentences. In the corpus, sentence segmentation is done automatically, based on silence detection. The average sentence length is 7.6 words. The sentences are orthographically transcribed and morpho-syntactically tagged. In addition, a complete and corrected syntactic dependency tree is built manually for each utterance.

Figure 1 contains an example sentence from the CGN corpus with the complete morpho-syntactic analysis. Note that certain leaves are not incorporated in the syntactic analysis tree. By definition we consider all those as disfluencies. In Figure 1 we have three disfluent chunks: a false start (ik uh), a filled pause following the word "scepsis" (uh) and a repetition (zo'n). According to our criterion, 9.07% of all lexical tokens in the data set are part of a disfluent chunk. The number of disfluent chunks is 27,113.

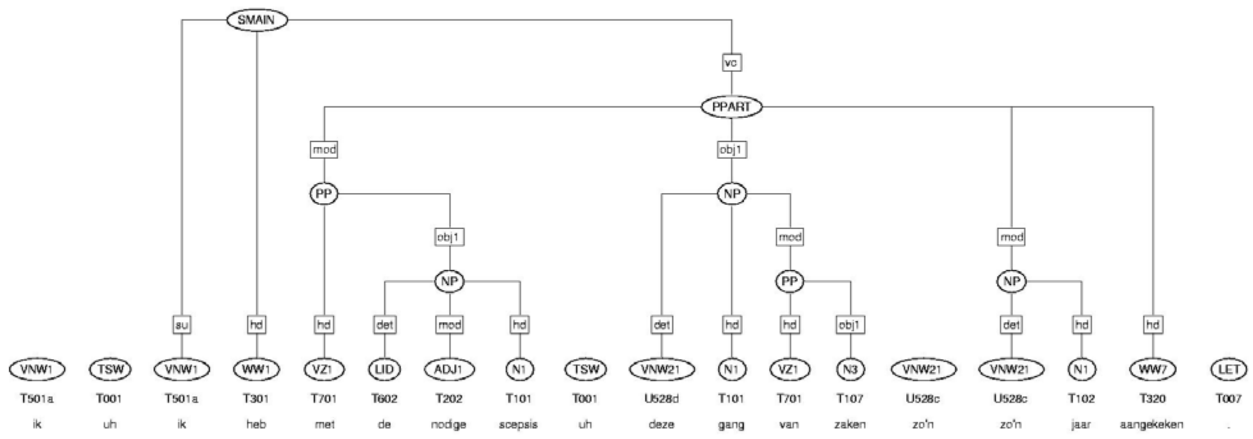

Figure 1: Example sentence ("ik uh ik heb met de nodige scepsis uh deze gang van zaken zo'n zo'n jaar aangekeken"; *I have followed this process with a certain amount of skepticism for about a year*) from the CGN corpus with full morpho-syntactic analysis.

2.2. Feature representations

Each of the 340,545 words is represented as a vector (or an *instance*) of 31 features that we extracted automatically from the corpus. The set of features can be grouped into two. One group consists of nine lexical string features that represent the focus word itself, plus its four left and right neighbors. Thus, we use a context window of length nine. In line with earlier work on disfluencies, for instance Heeman & Allen [7], we assume that local context is sufficient for detecting most speech repairs. We do not use part-of-speech tags and other syntactic information from the gold-standard corpus, nor from any part-of-speech tagger or parser. The second group of features consists of 22 binary overlap features. Of these, 20 record overlap between words within the window, the remaining two record overlap in the initial letters between the focus word and its left and right neighbors respectively. Matching words or word-initial letters are often to be found at the onset of a reparandum and/or a repair part of a disfluency.

Finally, for each word in the data set we record whether it is *inside* a disfluent chunck (I-DISFL) or *outside* of it (O-DISFL), i.e., whether it is part of the syntactic structure for the entire utterance or not. This is the class to be predicted by the machine learner.

2.3. Classification: Memory-based learning

We worked with a memory-based learning (MBL) algorithm based on the classical *k*-nearest neighbor approach to classification (Cover & Hart [3]). The *k*-NN algorithm looks for those instances among the training data that are most similar to the test instance according to some distance function Δ between two instances *X* and *Y*,

$$\Delta(X,Y) = \Sigma^{n}_{i=1}\, w_i\, \delta\, (x_i, y_i),$$

where n is the number of features in X and Y, w_i is the weight of feature i and δ gives the difference between two values of the ith feature. The classes of the k nearest neighbors are then extrapolated to predict the test instance's class. Memory-based learning is often called "lazy" learning, because the classifier simply stores all training data in memory, without abstracting away from individual instances in the learning process. We use the TiMBL 4.3 software package (Daelemans et al. [4]) for the experiments.

2.4. Experimental set-up

Training and testing is done by 10-fold cross-validation (CV), where re-sampling draws on discourse-based partitioning, thereby assuring that no material from the same discourse could be part of both the training and the test set.

The performance of the learner is evaluated in terms of four measures: *accuracy* (the overall percentage of correctly predicted I-DISFL and O-DISFL class labels), *precision*, *recall* and *F-score*. The F-score represents the harmonic mean of precision and recall. We use the unweighted variant of the F-score that is defined as $2PR/(P+R)$, where P is precision and R is recall (see e.g., van Rijsbergen [15]). We would like to stress that precision, recall and F-score apply to entire chunks in our evaluation. Thus: in the example sentence in Figure 1 *both* words in "ik uh" need to be classified as I-DISFL to count as a correct classification of the chunk.

2.5. Attenuation

Infrequent or unknown words are often problematic for machine learning techniques since the occurrence statistics of such items are unreliable. At the same time, the word form may contain useful information; for instance, a capitalized word is likely to be a named entity, a word that contains a number is usually either a number or the name of a number, a hyphen tends to indicate compounding. In addition, the final letters of a word may give away morphological clues, e.g., -ly (adverb) in English, or -dt (verb) in Dutch. Attenuation is a technique for words occurring below a certain frequency threshold to make such information explicit while masking the actual expression. Besides addressing the sparse data problem, another advantage of this technique is that the search space is reduced since the number of different feature values that needs to be checked becomes much smaller.

Our attenuation method is a simplified version of van den Bosch & Buchholz [2] (which was in turn based on a proposal by Eisner [5]):

```
IF a word occurs less than 100 times in the training data,
THEN
    • Convert it to MORPH
    • If it contains a number, add –NUM
    • If it contains a hyphen, add –HYP
    • If its first letter is a capital, add –CAP
    • If none of these three tests apply, add the last two
      letters of the word.
ELSE retain the original word.
```

For the (English) example sentence in Figure 1 this strategy produces the sequence "I have MORPH-ed this MORPH-ss with a certain amount of MORPH-sm for about a year". The attenuation method is applied to each train/test split, creating attenuated versions of both. We hypothesize that for the current learning task attenuation will not have a negative effect (and might even have a positive effect) since the binary overlap features, which are not based on the attenuated words, are likely to compensate for some of the potential information loss.

2.6. Parameter optimization by iterative deepening

Like most other machine learning techniques, the MBL algorithm has various parameters that may bias its performance. Since it is unknown beforehand which parameter setting is most likely to yield the best results, and since it would be bad practice to make this estimation using the test data, we performed parameter optimization experiments on the training material itself. More precisely, we ran 10-fold CV experiments on each of the 90% training sets within the basic 10-fold CV experiment. Parameter settings were tested according to a procedure called *iterative deepening*, which is a combination of classifier wrapping and progressive sampling (Kohavi & John [9], Provost et al. [14]).

The iterative deepening search algorithm automatically constructed a large number of different learners by varying the parameters of MBL and systematically trained these learners on portions of the 90% training set, starting with a small sample and doubling it over the iterative optimization rounds. In the iterating rounds of the search process the combinations of parameter settings were recursively estimated by maximizing the F-score performance on the I-DISFL class. The learner with the highest F-score on disfluency chunking (i.e., the one with the highest estimated generalization performance) was then selected and applied to the full 90% training set and subsequently tested on the as yet unseen 10% test set.

The learners were created by combining the following parameters (the default setting in TiMBL are shown in brackets):

- The value of k could be 1, 3, 5, 7, 9, 11, 13, 15, 19, 25, or 35. *(Default: 1)*
- The distance weighting metric Δ was majority class voting, linearly inversed distance weighting, inverse distance weighting, or exponential decay distance weighting with α set to 1, 2 or 4. *(Default: majority class voting)*
- The distance δ between feature values was computed using either the overlap function or the modified value difference metric. *(Default: overlap)*
- The weighting function w —which estimates the importance of attributes— was either information gain, gain ratio, χ^2, shared-variance or no weighting. *(Default: gain ratio)*

For details about the parameters see Daelemans et al. [4].

2.7. Baselines

To quantify the performance of the learning method we need to define a baseline. The most straightforward baseline is to always predict the majority class. Since most words in the corpus are not disfluencies, this baseline amounts to always predicting class O-DISFL. This would result in a correct prediction in 90% of the cases. However, for the class of interest (I-DISFL) this strategy leads to a recall of 0 (all disfluencies are missed) and an undefined precision and hence an undefined F-score.

Table 1: Majority class and filled pause (FP) baselines. Standard deviations are given between brackets.

	Acc.	Prec.	Rec.	F
Majority baseline	90.0 (1.6)	N/A	0	N/A
FP baseline	92.9 (1.5)	76.4 (3.1)	28.5 (5.8)	41.2 (6.7)

A somewhat more intelligent baseline is the following. The most frequent kind of easily detectable disfluencies are basic filled pauses (FPs, transcribed as uh, uhm, hu, and hm in the CGN corpus). We define a FP-baseline that predicts that all filled pauses are disfluencies and everything else is not. This baseline has an accuracy of 92.9%, a relatively high precision (not 100%, since one in four filled pauses is part of a larger disfluent chunk), a relatively low recall (it misses most disfluent chunks) and an overall F-score of 41.2%. Table 1 summarizes the baselines.

3. Results

Table 2 shows the average performance of MBL in three series of 10-fold CV experiments. In the first series we tested the default settings of the TiMBL implementation of MBL. This resulted in a 95.7% accuracy and a 72.3% F-score, which is a clear improvement of both baselines in Table 1. When the default settings of TiMBL were applied to the attenuated data, we observed a slight, overall improvement. The increase in accuracy is not significant, but the increase in F-score is, on a one-tailed t-test (t = 1.96, p < .05). Thus, attenuation indeed does not degrade performance while reducing the number of different feature values.

Table 2: Results of the three series of learning experiments. Standard deviations are given between brackets.

	Acc.	Prec.	Rec.	F
Default MBL	95.7 (0.5)	69.0 (3.0)	76.0 (1.6)	72.3 (1.8)
Attenuation + Default	96.0 (0.5)	71.7 (3.0)	76.3 (1.6)	73.9 (1.8)
Attenuation + Optimization	97.0 (0.5)	79.9 (3.1)	80.2 (5.8)	80.0 (1.8)

The third experiment series involved both attenuation and parameter optimization. This approach resulted in 97% accuracy in disfluency chunking, which is a substantial improvement over both baselines. With respect to the sharp FP-baseline this amounts to an error reduction of 58%. There is a 1-point increase in accuracy with this technique compared to the default learner applied to attenuated data. The difference is statistically significant (t = 4.29, p < .001). In addition, the F-score obtained in this experiment is almost twice as high as for the FP-baseline, and reaches a 7.7 points increase in F-score compared to the default learner, primarily due to an improved precision.

The settings resulting from the optimization process slightly differed for the ten folds. Namely, the optimal value of k ranged from 11 to 35, the distance metric chosen was either linearly inversed or inverse distance weighting, the distance between feature values was best computed using the modified value difference metric, and for feature weighting the learners mainly used shared variance, although gain ratio for two folds led to better results. The most reliable features for the learners were the focus word itself, as well as information on the focus word's overlap with the immediate right or second right word in the context window.

4. Discussion

We set out to investigate the usefulness of memory-based machine learning techniques for automatic disfluency chunking in transcriptions of spontaneous speech. We took a broad conception of what counts as a disfluency, namely everything that does not fit under the syntactic tree of a sentence according to a human annotator. This includes, among other things, filled pauses, false starts, repetitions, abandoned constituents, and fragmented words. We extracted simple, low level features that keep track of the local context, the focus word, and potential overlaps between words in the context window. It turned out that the best results were obtained using attenuated data and iterative deepening parameter optimization. We saw that optimization led to a significant improvement over the baselines and over the results obtained with the default TiMBL settings, yielding an accuracy of 97% (an error reduction of almost 60% with respect to the highest baseline) and an F-score of 80%.

An obvious limitation of the current study is that it is based on orthographic (correct) transcriptions. It would be highly interesting to see what happens if we first put the speech data through an automatic speech recognizer, and perform the learning experiments on its output (which is more than likely to contain a lot of recognition errors). It seems a safe bet that this will lead to a significant drop in performance. However, we believe that the basic approach followed in this paper will still be useful. For instance, overlap features may still be informative, even if entire words are misrecognized.

In addition, we conjecture that the use of additional features may compensate for some of the loss in performance. This would hold in particular for prosodic features, which have been shown to be indicators of certain kinds of disfluencies. One portion of the CGN data used here is currently being prosodically annotated, assigning pitch accents and breaks to the corpus material. We plan to redo the machine learning experiments described here using recognized words and prosodic information, and hope to be able to report on the results in a sequel to this paper.

5. References

[1] Bear, John, John Dowding & Elizabeth Shriberg. 1992. Integrating multiple knowledge sources for detection and correction of repairs in human computer dialog. *Proceedings ACL*, pp. 56–63.

[2] van den Bosch, Antal & Sabine Buchholz. 2002. Shallow parsing on the basis of words only: A case study. *Proceedings ACL*, pp. 433–440.

[3] Cover, Thomas & Peter Hart. 1967. Nearest neighbor pattern classification, *IEEE Transactions on Information Theory*, vol. 13, pp. 21–27.

[4] Daelemans, Walter, Jakub Zavrel, Ko van der Sloot & Antal van den Bosch. 2002. TiMBL: Tilburg memory based learner, version 4.3, reference guide, Tilburg University, available from `http://ilk.uvt.nl`.

[5] Eisner, Jason. 1996. *An empirical comparison of probability models for dependency grammar*. Technical report IRCS 96–11, Institute for Research in Cognitive Science, University of Pennsylvania.

[6] Eklund, Robert & Elizabeth Shriberg. 1998. Crosslinguistic Disfluency Modelling: A Comparative Analysis of Swedish and American English Human–Human and Human–Machine Dialogues. *Proceedings ICSLP*, Sydney, vol. 6, pp. 2631–2634.

[7] Heeman, Peter & James Allen. 1994. Tagging speech repairs. *ARPA Workshop on Human Language Technology*, pp. 187–192, Princeton.

[8] Hindle, Don. 1983. Deterministic Parsing of Syntactic Nonfluencies. *Proceedings ACL*, pp. 123-128.

[9] Kohavi, Ron & George John. 1997. Wrappers for Feature Subset Selection, *Artificial Intelligence*, vol. 97(1–2), pp. 273–324.

[10] Lendvai, Piroska. 2003. Learning to identify fragmented words in spoken discourse, *Proceedings EACL-2003 Student Research Workshop*, pp. 25–32.

[11] Nakatani, Christine & Julia Hirschberg. 1993. A Speech-First Model for Repair Detection and Correction. *Proceedings ACL*, pp. 46–53.

[12] Oostdijk, Nelleke. 2002. The design of the Spoken Dutch Corpus. In: *New Frontiers of Corpus Research*, P. Peters, P. Collins & A. Smith (eds.), Rodopi, Amsterdam, pp. 105–112.

[13] Oviatt, Sharon. 1995. Predicting spoken disfluencies during human-computer interaction, *Computer Speech & Language*, vol. 9, pp. 19–36.

[14] Provost, Foster, David Jensen & Tim Oates. 1999. Efficient Progressive Sampling, *Knowledge Discovery and Data Mining*, pp. 23–32.

[15] van Rijsbergen, Keith. 1979. *Information Retrieval*. Buttersworth, London.

[16] Shriberg, Elizabeth, Anderas Stolcke & Don Baron. 2001. Can prosody aid the automatic processing of multi-party meetings? Evidence from predicting punctuation, disfluencies and overlapping speech. *Proceedings ISCA workshop on Prosody in Speech Recognition and Understanding*, pp. 139–146.

[17] Spilker, Jörg, Anton Batliner & Elmar Nöth. 2001. How to repair speech repairs in an end-to-end system. *Proceedings of Disfluency in Spontaneous Speech*, pp. 73–76.

[18] Tjong Kim Sang, Erik & Sabine Buchholz. 2000. Introduction to the CoNLL 2000 shared task: Chunking. In: *Proceedings of CoNLL-2000 and LLL-2000*, pp. 127–132.

[19] Wouden, Ton van der, Heleen Hoekstra, Michael Moortgat, Bram Renmans & Ineke Schuurman. Syntactic Analysis in the Spoken Dutch Corpus. In: M. González Rodriguez & C. Paz Suárez Araujo, *Proc. Third International Conference on Language Resources and Evaluation*, pp. 768–773.