Exploring Features and Classifiers for Dialogue Act Segmentation

Harm op den Akker[1] and Christian Schulz[2]

[1] Twente University, Enschede, The Netherlands
[2] Deutsche Forschungszentrum für Künstliche Intelligenz (DFKI), Saarbrücken, Germany

Abstract. This paper takes a classical machine learning approach to the task of Dialogue Act segmentation. A thorough empirical evaluation of features, both used in other studies as well as new ones, is performed. An explorative study to the effectiveness of different classification methods is done by looking at 29 different classifiers implemented in WEKA. The output of the developed classifier is examined closely and points of possible improvement are given.

1 Introduction

Current research in the AMIDA project is focussed on a deeper understanding of meeting discourse semantics. An important first step towards this goal is the structuring of utterances into meaningful parts, like sentences or Dialogue Acts. A Dialogue Act is a sequence of subsequent words from a single speaker that form a single statement, an intention or an expression. Segmenting spoken or written text into Dialogue Acts contributes to a better understanding of the utterances; is the speaker for example asking something, or is he conveying a meaning? A lot of work on DA and linguistic segmentation and DA classification (or tagging) has already been done, on the AMI Corpus: [1], but also on other corpora [2], [3], [4], [5] and [6]. However, this is not a closed topic yet and many techniques still need to be explored.

The following is a Dialogue Act annotated example taken from the AMI corpus, using its 15 tag tagset:

You know. **(Elicit-Assessment)** *Yep.* **(Assess)** *Mm-hmm.* **(Backchannel)** *I think one factor would be production cost.* **(Inform)**

This is an example of a segmentend and tagged series of words. This paper looks at the task of segmentation alone, where the input is the sequence of words (without interpunction or capitalization), and the goal is to tag each word as either a segment **S**tart or an **I**nternal:

you/**S** *know*/**I** *yep*/**S** *mm-hmm*/**S** *i*/**S** *think*/**I** *one*/**I** *factor*/**I** *would*/**I** *be*/**I** *production*/**I** *cost*/**I**

The Dialogue Acts as used within the AMI project are defined to be uttered by a single speaker only (no single Dialogue Act can span multiple speakers). For this reason, the input for the classifier is the sequence of words sequentially uttered per speaker.

This paper looks into stand-alone segmentation using a two-class classification approach. It can also be seen as a sequence-based machine learning problem in which case techniques like Hidden Markov Models, Conditional Random Fields (CRFs) or Memory-Based Tagging can be used.

For every machine learning task there are four major aspects that can and should be looked into. These are: features, classification method, classifier parameters and data. This article will focus on the evaluation of features using a Bayesian Network classifier, and takes a quick look at other classification methods.

For this project the AMI Corpus is used, which will be described first. Next, a list of features that have been derived from the data is defined. All these features need to be evaluated to see how useful they are. Simply using all available information to train classifiers is not efficient. The amount of training data needed for the classifier to automatically learn that some features, for example, contain little or no information can become extremely large. Therefore a thorough feature selection is conducted, which is described in Section 4. Experiments to explore the influence of different classification techniques are described 5. Then, a thorough evaluation of the final results (for Bayesian Networks) will be done, which gives insight in the types of errors made by the classifier. This evaluation also shows that using standard performance measures like precision and accuracy should not be taken too strict because of the large amount of non-harmful errors that are made.

2 The AMI Corpus

All the data that is used in this project comes from the AMI Corpus [7]. The largest part of the corpus (72 hours in total) are scenario-based meetings which are covered by 35 series, totalling 138 meetings. The following split into training-, test,- and evaluation sets has been made:

Training set: ES2002, ES2005-2010, ES2012, ES2013, ES2015, ES2016, IS1000-1007, TS3005, TS3008-3012 (98 meetings)
Evaluation set: ES2003, ES2011, IS1008, TS3004, TS3006 (20 meetings)
Development set: ES2004, ES2014, IS1009, TS3003, TS3007 (20 meetings)

The training set contains 465.478 words, the evaluation set 106.742 words and the development set contains 99.372 words [3]. For all the words, begin- and endtime information is available as well as the person who uttered the word and his/her role in the meeting (either Project Manager, Industrial Designer, Marketing Expert or User Interface designer). Also the wave signals of all the meetings and for every participant are available, which will be used for prosodic feature extraction. All annotations are hand transcribed.

[3] For this research, the evaluation-, and development sets are switched around from the standard AMI configuration.

3 Definition of Features

The input for the classifier is a feature-vector for each word in the corpus. Features are derived from the word itself, timing and prosodic information. The types of features that have been used in other work on DA segmentation will be used, as well as some features that are introduced here.

Next follows a list of all the features used in this project. The number ($\#x$) is used throughout the article to identify the feature.

3.1 Time related features

Features derived from the start- and end times of the words in the corpus:

- **#1. Pause between the words:** the duration between the starttime of the current word and the endtime of the last word by the same speaker.
- **#2. Pause nominal:** a 'boolean feature', *pause* or *no pause*. Note that due to forced time alignment in the AMI corpus, all small pauses between words have been truncated to zero.
- **#3. Duration of word:** self-explanatory.
- **#4. Mean duration of word:** the average duration of this word in the whole training set.
- **#5. Relative duration of word:** the duration of the word minus the mean duration of the word.

3.2 Word related features

Features derived from the words themselves.

- **#6. Current word:** the word itself. Because most classifiers cannot handle String input, the feature is converted into a nominal feature for each word in the corpus (e.g. 'current_word_hello' {Yes,No}). Only words that occur more than 100 times are considered (452 in total)[4].
- **#7. Next word:** the next word from the same speaker (with frequency of 100 or more).
- **#8. Previous word:** the previous word from the same speaker (with frequency of 100 or more).
- **#9. Part-of-Speech Current word:** a part of speech tag given to the current word. The word is tagged by the Stanford Part-Of-Speech Tagger[5], where the input to the tagger is the current word and the surrounding 6 words. This window of 7 words is also used in e.g. [8]. The tagger uses the Penn Treebank English tagset, which is a commonly used tagset consisting of 37 tags [9].

[4] Experiments have shown that less frequently occuring words contain no information on segment boundaries.

[5] http://nlp.stanford.edu/software/tagger.shtml

- **#10. PoS Previous word:** the Part-Of-Speech tag for the word preceding the current word (Penn Treebank Tagset).
- **#11. PoS Next word:** the Part-Of-Speech tag for the word following the current word (Penn Treebank Tagset).
- **#12. #13. #14. PoS Reduced Tagset:** intuitively, a 37-tag tagset is too fine grained for the DA segmentation task. The Penn Treebank set has been mapped to a 6 tag set: *Verbs*, *Nouns*, *Adjectives*, *Adverbs*, *WH-words* and *Other*. Feature #12 is the current PoS tag using this reduced tagset, feature #13 the PoS of the previous word and feature #14 that of the next word.
- **#15. #16. #17. PoS with keywords:** these three features, for the current word, previous word and next word respectively, use the reduced tagset but extended with certain important keywords. This approach is analogous to [3], where some words get their own tag. Preliminary experiments show that the most important cue words for a segment border are: ‘*yeah*’, ‘*so*’, ‘*okay*’, ‘*but*’ and ‘*and*’, so these get their own tag.
- **#18. Word repeat:** true, if the current word is the same as the next word, false otherwise.
- **#19. Word repeat 2:** true, if the current word is the same as the previous word, false otherwise.

3.3 Prosodic features

Features derived from the word pitch and energy information[6]. All values have been normalized to the microphones.

- **#20. #21. #22. Pitch Features:** the minimum-, maximum- and mean pitch respectively.
- **#23. #24. #25. Energy Features:** the minimum,- maximum- and mean energy respectively.
- **#26. Speechflow Past:** this feature defines a ‘talking speed’ over the current word W_0 and the words W_{-1}, W_{-2} and W_{-3}. The feature is the total time of those 4 words (including pauses in between them), divided by the total number of syllables in the words.
- **#27. Speechflow Future:** the talking speed over the future 3 words: the time of words $W_0 - W_3$ divided by the total number of syllables in those words.
- **#28. Speechflow Change:** substract feature #27 from feature #26.

3.4 Online features

The following four features must be calculated during the classification because their value depends on previously assigned borders.

- **#29. Number of words in previous segment:** self-explanatory.

[6] Thanks to Gabriel Murray for supplying the scripts and to Sebastian Germesin for the word-alignment

– **#30. Distance (number of words) to the last segment:** a counter that keeps track how far this word is away from the last assigned border.
– **#31. Relative position of word inside segment:** analogous to feature #30 only now the counter counts blocks of 5 words, see [10].
– **#32. Time interval of current word to last segment:** the total time between the end of the last segment border and the beginning of the current word.

4 Feature Selection Results

The goal of the feature selection phase is to see what the influence of individual features is, as well as to see how certain combinations of features influence the classifier performance. There are 32 different features, three of which (the word features) are expanded to one feature for every frequently occuring word. This makes the total feature vector 1309-dimensional. To find an optimal feature subset within this space is computationally impossible (2^{1309} possibilities), so the first step is to filter out those features that contain little information. To do this, the InfoGain Attribute Evaluator from WEKA[7] is used. This method calculates the probability of an instance being a segment border (prior probability) and compares this to the probability of a segment border given that a feature has a certain value. The higher the change in probability, the more useful this feature is. The results of the InfoGain ranking top 30 can be seen in Table 1.

Table 1. Result of the Information Gain ranking algorithm on all features.

Rank	Feature	Infogain	Rank	Feature	Infogain
1	#1	0.2462	**16**	#8 (yeah)	0.0260
2	#2	0.2399	**17**	#11	0.0213
3	#4	0.2191	**18**	#27	0.0190
4	#5	0.1373	**19**	#6 (so)	0.0152
5	#26	0.1311	**20**	#30	0.0152
6	#32	0.1066	**21**	#6 (okay)	0.0139
7	#15	0.0818	**22**	#8 (okay)	0.0114
8	#16	0.0664	**23**	#25	0.0107
9	#9	0.0517	**24**	#17	0.0106
10	#10	0.0490	**25**	#6 (but)	0.0102
11	#28	0.0461	**26**	#6 (and)	0.0088
12	#13	0.0424	**27**	#12	0.0086
13	#6 (yeah)	0.0347	**28**	#21	0.0084
14	#29	0.0323	**29**	#6 (mm-hmm)	0.0084
15	#23	0.0308	**30**	#6 (mm)	0.0078

In the second step of the feature selection phase, a BayesNet classifier has been repeatedly trained and evaluated with different feature subsets, taken from

[7] http://www.cs.waikato.ac.nz/ml/weka/

the best 30 features (Table 1). Because an exhaustive subset evaluation is still impossible to do, feature subsets are manually chosen and expanded upon, until all options that are likely to improve the results have been tried. A total of 528 experiments have been done, and the resulting best performing subset is seen in Table 2.

Table 2. Best Performing Feature Subset for Bayesian Network Classifier.

ID	Feature
#1	Pause between the words
#2	Pause nominal
#4	Mean duration of word
#6	Current word (mm-hmm, but, yeah, so, okay, and)
#8	Previous word (okay)
#10	PoS Previous word
#11	Pos Next word
#16	Pos Previous word (with keywords)
#23	Minimum Energy
#25	Mean Energy
#28	Speechflow change
#32	Time interval of current word to last segment

This feature set achieved an F-measure of **0.75** with **0.79** precision and **0.72** recall. There are a few interesting things about this set. First, the *pause* and the *nominal pause* (#1 and #2) seem to contribute both, even though they are obviously correlated. Leaving one of them out lowers the performance of the classifier. Second, the Part-of-Speech tags of the previous and next word (#10 and #11) are selected, but not that of the current word (this doesn't seem intuitive). Third, in contrast to the literature, the words of the Part-of-Speech tagset enhanced with keywords (#15, #16 and #17) are better added as individual features, instead of incorporating it in a PoS-feature like in [3].

5 Classifier Experiments

Bayesian Networks are just one of many possible techniques for building a machine classifier. To get a quick overview of how other classifiers handle the task of DA segmentation, a number of different classifiers have been trained and tested. An exhaustive search in the feature - classifier - parameter space is extremely time consuming, therefore we save time by a) using default classifier parameters and b) use only 50.000 training instances[8]. Note that different classifiers may be affected differently by size of training set, parameter optimization and feature sets, so optimally these should all be varied [11]. In these experiments, the

[8] Experiments showed that after 50.000 instances, no significant improvements in results could be noticed with a Bayesian Network classifier

97 best performing feature sets[9] have been fed to the different classifiers. The results are reported for the best feature vector (F-measure) for every classifier (see Table 3). The names of the classifiers in Table 3 refer to the names of the WEKA classes that implement them

Table 3. Classifier experiment results.

Classifier	F-measure	Classifier	F-measure
LMT	0.76	DecisionTable	0.73
J48	0.75	VotedPerceptron	0.73
NBTree	0.75	SimpleLogistic	0.73
ADTree	0.75	SMO (Poly)	0.73
SimpleCart	0.75	SMO (RBF)	0.73
PART	0.75	Ridor	0.73
BayesNet	0.75	RBFNetwork	0.73
REPTree	0.75	NNge	0.70
...	...	...	...
Logistic	0.73	HyperPipes	0.27

Many of these classifers perform in line with the Bayesian Network classifier, whereas the variance in results is much larger when changing the feature set. This shows that changing the classification method is far less significant than optimizing features. It is unclear whether any of the classifiers score significantly better than the BayesNet classifier, but because this is only an exploratory experiment we don't worry about that now and continue the experiments using Bayesian Networks.

6 Evaluation

In order to determine how good the results are, they are compared to a baseline and a theorized top score, both based on the intrinsic properties of the task. The baseline is defined by a least-effort method. The 'maximum achievable score' is based on the intrinsic vagueness of the Dialogue Act problem by looking at the inter-annotator confusion analysis of Dialogue Act segmentation in [1]. The least effort, or baseline classifier consists of a single rule: *if there is a pause between two words, the second word is the start of a new Dialogue Act.* See Table 4 for a summary of the results [10].

To answer the question how good a 0.05 improvement on the baseline is, we must hypothesize a roof for the results. The best score we can expect from an automatic Dialogue Act segmenter depends on how "vague" a segment boundary

[9] All scoring better than pause feature alone.

[10] The added "NIST Sentence-like Units" error metric is used in many other publications. It is the total number of false positives and false negatives, divided by the total number of boundaries in the reference data.

Table 4. Result overview for evaluation- and test set, including baseline, using BayesNet classifier.

Set	Instances	Acc.	Prec.	Recall	F	NIST-SU
Development	99372	0.93	0.79	0.72	0.75	0.47
Evaluation	106742	0.92	0.79	0.72	0.76	0.47
Baseline	106742	0.92	0.97	0.55	0.70	0.47

is defined, or what the intrinsic difficulty of the task is. As a best possible score we take the results from [1] on inter-annotator confusion analysis. Table 5 shows the recall/precision values between two different annotators if one is taken as the gold-standard and the other as the “classifier output”. The table is based on the IS1003d meeting which has been annotated by four different annotators.

Table 5. Recall/Precision and average F-score values for inter-annotator segmentation.

	dha	dha-c	mar	s95	s95-c	vka	**Avg. F-score**
dha	–	0.91	0.93	0.84	0.92	0.91	**0.86**
dha-c	0.93	–	0.94	0.84	0.92	0.91	**0.87**
mar	0.77	0.79	–	0.76	0.80	0.86	**0.85**
s95	0.81	0.83	0.89	–	0.85	0.87	**0.82**
s95-c	0.89	0.91	0.94	0.85	–	0.92	**0.87**
vka	0.72	0.74	0.83	0.72	0.75	–	**0.82**
Total average F-score:							**0.85**

The last column for each row contains the average F-measure for the annotator on that row with all the other annotators. The total average F-score for all annotator pairs is 0.85. This can be seen as a maximum achievable score. Note that the argument that a classifier is capable of learning through noise, and thus perform better may be true, but this is not reflected in F-score. Most importantly, this puts our results of around 0.76 into perspective: on a scale from 0.70 to 0.85, we're achieving only one third of what is possible!

6.1 Detailed error analysis

To pinpoint where we can improve the performance of our classifier, and see how representative the F-score measure is for this taks, we take a look at the output it produces. Because the definition of a Dialogue Act is not perfectly defined, some of the output may mismatch that of the gold standard, but could still be considered correct. Table 6 lists the words that are most frequently incorrectly classified.

It is interesting to see that the words ‘*yeah*’, ‘*so*’, ‘*okay*’, ‘*but*’ and ‘*and*’ that have previously been proved to be useful features, now all occur in the top 6 of most frequently occuring errors. They all have word-error percentages between 20% and 35% and the six words make up 42% of the total amount of errors made

Table 6. Word error distribution (evaluation set).

word	occurence	total errors	error percentage	false positives	false negatives	true positives	true negatives
and	2160	761	35%	531 (24%)	230 (11%)	719	680
yeah	3437	675	20%	629 (19%)	46 (1%)	2640	122
i	2471	552	22%	82 (3%)	470 (19%)	472	1447
so	1667	470	28%	395 (24%)	75 (4%)	1003	194
okay	1303	410	31%	393 (30%)	17 (1%)	823	70
but	975	322	33%	274 (28%)	48 (5%)	603	50
um	1205	290	24%	126 (10%)	164 (14%)	375	540
uh	2980	268	9%	42 (1%)	226 (8%)	333	2379
you	2359	219	9%	18 (0%)	201 (9%)	118	2022
we	1677	196	12%	9 (1%)	187 (11%)	72	1409
it's	1076	177	16%	29 (2%)	148 (14%)	166	733
that's	727	163	22%	42 (5%)	121 (17%)	112	452
the	4753	145	3%	13 (0%)	132 (3%)	121	4487
or	681	137	20%	17 (2%)	120 (18%)	57	487
it	2157	117	5%	1 (0%)	116 (5%)	85	1955

by the classifier. The first 20 words cover 70% of the total errors, while the other 30% is covered by 507 other words. This shows that a large improvement can be gained by looking at a small number of words. Therefore, we take a closer look at the 6 words that cause 42% of the errors. We distinguish between errors that are not actually harmful, like not splitting up an "*um - yeah*" into two segments, and *real errors* like not to make the split in "*...we'll discuss that - and then I just wanna mention some new project requirements...*". For the following 6 words, the results of 100 false positive, and 100 false negative error cases were closely examined. For every word, some basic cases or rules are identified that could help in improving the segmenter.

The 'and' case: For the false positives, 29% of the errors were not considered harmful. All non-harmful cases can be described as belonging to the following class:

– **Disfluency class:** cases were the word is preceded by a disfluency ("um", false start, etc...). The classifier and gold standard segmentation often do not agree on whether the disfluency is part of the previous or the next segment (or is a segment in itself). For example: "...I mean fr - and from the point of view ...", where the classifier seperates the false start, but the annotator did not.

The remaining real errors largely corresponded to one of the following two classes:

- **‘And such and so’ class:** cases of “... and such”, “... and so on”, “... and stuff”. In these cases, there probably should not be a boundary.
- **‘Fruit and Vegetables’ class:** most of the false positives are related to splitting a summation of items into two segments, like: “[research] and [development]”, “[my brother] and [my dad]”, “[up] and [down]”. But some of the examples are a bit more complex, like: “[the actual lcd] and [maybe to a certain extent the joystick]”.

The ‘yeah’ case: Looking at the examples for ‘yeah’ is unfortunately quite uninformative most of the times.

The ‘I’ case: Approximately one third of the FP’s can be seen as non-harmful and can be categorized in the same “disfluency class” as for the ‘and’ cases. For the real errors, about 55% of the errors belong to this class:

- **‘Yeah I class’:** an ‘I’ following a ‘yeah’ or sometimes an ‘okay’, as in “...yeah I think so”. There is no pause between the ‘yeah’ and the ‘i’, so the ‘yeah’ is not just a short backchannel, but part of the statement. In this case the ‘I’ should not be tagged as boundary.

The ‘so’ case: The FN’s can sometimes be considered non-harmful because they are close to “um’s”. For the FP’s, 47% of the errors can be seen as non-harmful and can be attributed to the “disfluency class”. For the actual errors, there are a few cases that could be handled differently:

- **non-Consequently class:** The word ‘so’ is often used as a conjunction, like: “...so a small speaker you mean...”. These cases are likely DA segment boundary candidates. But the word ‘so’ can also be used in other cases like “...so far so good...”, “...i think so...”, or “...that’s so great...”. In these cases it is far less likely to be a segment boundary.

The ‘okay’ case: Half of the FP’s can be subcategorized in the following two classes:

- **Double-positive class:** 24% of the FP’s are examples where the classifier splits a double-backchannel or “positive expression” into two, like: “...yeah okay...”, “...right okay...” or “...okay okay...”.
- **Uhm-okay class:** 18% of the false positives are cases where a sort of ‘uhm’ preceding an ‘okay’ is split in two; for example: “...oh okay...’, “...uh okay ...” or “‘...hmm okay”. These are considered non-harmful errors.

The ‘but’ case: For the FN’s, 9 out of the 48 errors can not really be considered harmful, because the ‘but’ has no real meaning. It is used as a filler/disfluency, where sometimes it is considered as a seperate segment, and sometimes it is part of the previous or next segment. For the FP’s, 27% can be seen as “disfluency class”, while 21% can be categorized as follows:

– **Yeah but class:** For example a false split between "...yeah - but...", "...no - but..." or "...okay - but...". In these cases there should generally be no boundary.

The detailed analysis of these error cases could be used to create a rule-based pre- or post processing system to aid the automatic segmentation methods. A lot of errors are produced by disfluency, so a preprocessing step to remove disfluency could also help increase performance (see [12]).

7 Conclusion

The experiments done on DA segmentation on the AMI corpus show that reasonable results can be achieved using a variety of word related, time related, online-, and prosodic features. More importantly it is shown that there are still quite a lot of things to be done that can possibly increase performance. The classifier experiments in Section 5 indicate that a lot of different classifiers already perform well with default settings. Solving the search problem of a combined optimization of feature subsets and classifier parameters could possible lead to a significant improvement in results, as [11] points out.

Further improvements could be achieved by optimizing the feature representation. The Part-of-Speech feature, for example, has proven to be useful, even though the tag-set has not been changed for the specific task of segmentation. A detailed analysis of the Part-of-Speech of words near segment boundary could lead to a better tag-set, and could possibly improve the overall classifier performance. The same goes for all the numeric features like pause, "speechflow" and the prosodic features, where optimal binning configurations can be found using simple brute-force techniques, such as in [13].

The detailed analysis of frequently occuring errors in Section 6.1 could provide a basis for a rule-based pre-processing of the data. Because the six words mentioned in this section make up such a large amount of the errors produced by the classifier, more attention should definitely be put into handling these cases. Since the words occur so frequently, there is enough data to train classifiers specifically for these words. In combination with rules covering the identified error classes, some improvement of the overall results can be expected.

Another important conclusion that can be drawn from the error analysis is that disfluency in the spontaneous speech in the AMI corpus causes a lot of gold-standard errors. These errors are not always expected to be very harmful, but it is worth looking into a way of avoiding them. A preprocessing step to correct disfluency errors might be very helpful for these types of errors.

Besides the abovementioned points that still need to be addressed, future work on Dialogue Act segmentation should include features that look at the interaction between speakers, as well as multimodal features like gaze, gestures and movement.

Acknowledgement

This work is supported by the European IST Programme Project FP6-0033812. The main author's contributions where done while on a traineeship at DFKI, Saarbrücken, Germany. Thanks to Peter Poller, Tilman Becker, Sebastian Germesin, Rieks op den Akker, Dirk Heylen and Dennis Reidsma.

References

[1] AMIDA: Augmented Multiparty Interaction with Distance Acces Deliverable D5.2: Report on multimodal content abstraction. Technical report, Brno University of Technology, DFKI, ICSI, IDIAP, TNO, University of Edinburgh, University of Twente and University of Sheffield (2007)

[2] Zimmermann, M., Liu, Y., Shriberg, E., Stolcke, A.: Toward joint segmentation and classification of dialog acts in multiparty meetings. Proceedings of MLMI'05, LNCS 3869 (2006) 187–193

[3] Stolcke, A., Shriberg, E.: Automatic linguistic segmentation of conversational speech. In: Proc. ICSLP '96. Volume 2., Philadelphia, PA (1996) 1005–1008

[4] Kolář, J., Liu, Y., Shriberg, E.: Speaker adaptation of language models for automatic dialog act segmentation of meetings. (2007)

[5] Zimmermann, M., Stolcke, A., Shriberg, E.: Joint segmentation and classification of dialog acts in multiparty meetings. In: Proc. IEEE ICASSP. Volume 1., Toulouse, France (2006) 581–584

[6] Cuendet, S., Shriberg, E., Favre, B., Fung, J., Hakkani-Tur, D.: An analysis of sentence segmentation features for broadcast news, broadcast conversations, and meetings. In: Proceedings SIGIR Workshop on Searching Conversational Spontaneous Speech, Amsterdam, Netherlands (2007) 37–43

[7] McCowan, I., et. al.: The AMI meeting corpus. Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research (2005)

[8] Poel, M., Stegeman, L., op den Akker, R. In: A Support Vector Machine Approach to Dutch Part-of-Speech Tagging. Volume 4723. Springer Berlin / Heidelberg (2007) 274–283

[9] Marcus, M.P., Santorini, B., Marcinkiewicz, M.A.: Building a large annotated corpus of english: The penn treebank. Computational Linguistics **19**(2) (1994) 313–330

[10] Dielmann, A., Renals, S.: DBN based joint dialogue act recognition of multiparty meetings. In: Proc. IEEE ICASSP. Volume 4. (April 2007) 133–136

[11] Daelemans, W., Hoste, V., De Meulder, F., Naudts, B.: Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. In: Proceedings of the 14th European Conference on Machine Learning (ECML-2003). Lecture Notes in Computer Science 2837, Cavtat-Dubrovnik, Croatia, Springer-Verlag (2003) 84–95

[12] Germesin, S., Becker, T., Poller, P.: Hybrid multi-step disfluency detection. MLMI'08 (2008)

[13] Webb, N., Hepple, M., Wilks, Y.: Empirical determination of thresholds for optimal dialogue act classification (2005)