Evaluating Acoustic, Textual and Grammar Features for Alcohol Classification

Felix Neutatz[1,2], Dennis Schmidt[1,2], Moritz Teckenbrock[1,2] and
David Suendermann-Oeft[1,3]

[1]DHBW Stuttgart, Germany
[2]IBM Deutschland MBS GmbH, Ehningen, Germany
[3]ETS, San Francisco, USA*

Abstract: This paper evaluates the detection of alcohol intoxication in speech, using different classification approaches. It compares the classification based on acoustic, textual and grammar features as well as the combination of these three. Furthermore, it examines the influence of experiment and category dependency. The paper concludes that neither classification using textual features nor classification based on grammar features can challenge the baseline. Nevertheless, experiment and category dependency have a significant influence on the result.

1 Introduction

One major issue in road traffic is alcohol intoxication. Every day more than 40 persons in Germany could be saved from injuries or even death, if alcohol intoxicated people were stopped from driving [1].

An idea to tell whether or not somebody is intoxicated is to record an individual's speech and apply machine learning techniques for classification.

In 2013 a research team worked on classifying intoxication with the help of text features only. They presented an outstanding result of 89.4% for the unweighted average recall on picture description experiments [2]. Motivated by this outcome, our goal was to outperform this baseline by using more sophisticated classification methods combining textual and acoustic features. Since in a realworld scenario the textual transcription of the evaluated individuals' speech is not available, we also used the text produced by a speech recognizer.

However, it turned out that the former research team's approach was not scientifically sound due to several inconsistencies described later. Therefore, their experimental results could not serve as valid baseline.

All experiments described in this paper are based on the Alcohol Language Corpus (ALC) provided by the Ludwig Maximilians University of Munich. The corpus includes voice recordings of people in alcohol intoxicated and sober state [3][4].

Our partitioning of the corpus instances into training, development and test set is adopted from the Interspeech 2011 Speaker State Challenge [5] as shown in table 1.

In the following we elaborate on the deficiency of [2], present and a scientifically sound approach to applying text classification to the scenario at hand, experimental results, and conclusions.

*The described work was done when D. Suendermann-Oeft was at DHBW Stuttgart.

Table 1 - Partitions of ALC by IS2011 and our reduced version (red)

	NAL	NAL (red)	AL	AL (red)	total	total (red)
Train	3750	1885	1650	1595	5400	3480
Develop	2790	1421	1170	1131	3960	2552
Test	1620	1566	1380	1334	3000	2900
Train+Develop	6540	3306	2820	2726	9360	6032
Train+Develop+Test	8160	4872	4200	4060	12360	8932

2 Issues with the previous study

There are several shortcomings in [2] rendering its outcomes unreliable. The most substantial problem is the application of attribute selection to the whole data set (including the test set). This results in an overoptimistic outcome.

Moreover, the comparison between the performance of different machine learning algorithms (e.g. Logistic regression, Naïve Bayes, SMO) was limited to the classifiers' default settings rather than applying specific parameter tuning. In one case, 10-fold cross-validation was performed on the whole data set. As there is a random distribution of the samples in the different folds, genders are not balanced, and the tests are not speaker independent.

Another major issue in [2] is the ambiguous definition of alcoholization and non-alcoholization. In [2] the class "alc" (alcohol) includes all recordings of persons who consumed alcohol. All other recordings are annotated as "nonalc" (non-alcohol). The Interspeech 2011 Speaker State Challenge (IS 2011) requested to outperform the baseline provided by [5] where the class "alc" was determined by the value of the blood alcohol concentration (BAC). All intoxicated persons having a BAC greater than 0.05% were classified as "alc".

For these reasons the results in [2] are not comparable to the IS 2011 [5] baseline.

3 Corpus

For the research work in this paper we used the ALC, which consists of 39 hours of speech from 77 female and 85 male speakers [6]. However, we classified on a reduced data set that was introduced by the IS 2011 challenge. In order to provide a gender-balanced setup, the utterances of 8 male speakers were discarded from the original data set. The ALC consists of different categories which are listed in table 2. Each category consists of a number of experiments. The category "read command" contains the experiment, where the user has to read "Autobahn meiden" or "nächster Titel", for example.

Some of the experiments that were conducted to gather the data of the ALC were applied to intoxicated persons only. There are twice as many experiments for the non-alcoholized class as for the alcoholized one. Moreover, one of the experiments was not conducted for intoxicated persons. Thus, there are 32 experiments in total that are not available in both class states. This gives text classification an unfair advantage over acoustic classification. Therefore, we removed these instances from the original data set.

We trained models on the training set, tuned the parameters on the development set and built final models with optimal parameters on the merge of training and development instances. In [2] there is also research on subsets of the corpus, representing the different categories of the conducted experiments, e.g. a model specifically built for the category of describing pictures. Some of these subset models turned out to achieve superior performance results compared to the general corpus model. Accordingly, as described in [2] we also assessed the impact of category dependency.

Table 2 - Partitions of ALC by IS2011 of our reduced version (red.)

# Cat.	Description	Speech Type	# Train		# Dev		# Test		# TOTAL	
			NAL	AL	NAL	AL	NAL	AL	NAL	AL
LN	List numbers	read	325	275	245	195	270	230	840	700
LT	List tongue twister	read	65	55	49	39	54	46	168	140
LS	List spelling	read	65	55	49	39	54	46	168	140
RT	Read tongue twister	read	260	220	196	156	216	184	672	560
RR	Read command	read	260	220	196	156	216	184	672	560
RA	Read address	read	260	220	196	156	216	184	672	560
DQ	Dialog question	spontaneous	65	55	49	39	54	46	168	140
DP	Dialog picture descr.	spontaneous	65	55	49	39	54	46	168	140
MQ	Monolog question	spontaneous	65	55	49	39	54	46	168	140
MP	Monolog picture	spontaneous	130	110	98	78	108	92	336	280
EC	Elicited command	spontaneous	325	275	245	195	270	230	840	700

Table 3 - weka.filters.unsupervised.attribute.StringToWordVector configuration

Function	value
setWordsToKeep()	1000000
setIDFTransform()	false
setTFTransform()	false
setLowerCaseTokens()	true
setOutputWordCounts()	true
setMinTermFreq()	2
setNormalizeDocLength()	new SelectedTag(1,StringToWordVector.TAGS_FILTER)
setNGramMinSize()	1
setNGramMaxSize()	3
setUseStoplist()	true
setStopwords()	new File(ḧttp://members.unine.ch/jacques.savoy/clef/germanST.txt)̈
setStemmer()	new SnowballStemmer(g̈erman)̈

Accordingly, in order to get test results for every single category we reduced the training, development and test set to samples of the corresponding category. On these subsets we used the aforementioned approach to build, tune and test individual models.

4 Experiments

We ran all experiments with the Weka v3.7 [7], a data mining framework written in Java. It features a good number of pre-processing and classification algorithms. Moreover, we leveraged the Weka plugin for LibSVM [8], an open-source support vector machine toolkit.

4.1 Text Features

For text classification we created a bag-of-words model with the help of the Weka function StringToWordVector. This function provides several parameters to modify the bag-of-words model. We experimented with the degree of n-grams, case sensitivity, IDF/TF transformation, deletion of stop words, word stemming, normalization by document length and minimum term frequency. The best configuration we found is described in table 3.

As database for text classification we used manually transcribed recordings provided by the ALC. In order to apply text classification to real-world applications, speech recognition is required. We used PocketSphinx [9] (v0.7) as speech recognizer and trained two different language models — one for tuning and one for testing. The statistical language model, based on trigrams, was built with the language model tools (CMUclmtk) of the CMUSphinx toolkit [10]. For tuning we took all training, for testing both training and development samples to create

Table 4 - Performance of speech recognition model on the whole corpus

Model	WER
Train	42.9%
Train+Develop	38.1%

Table 5 - Baseline reproduction

Smote		SVM	Results	
Percentage	KNN	c	UAR dev	UAR test
127	5	0.02	65.3%	65.7%

the model. Moreover, we generated two corresponding word lists, which were converted to dictionaries by adding phoneme transcriptions. For this purpose we applied Sequitur G2P, a state-of-the-art grapheme-to-phoneme converter [11]. Since the G2P software utilizes a statistical approach to find the corresponding phonemes, we used the lexicon of the training set of the Verbmobil Corpus to train the G2P model [12].

We measured the performance of the speech recognizer for all available samples (15,180) of the corpus shown in table 4. In addition to the text features which were recognized by the ASR, we also added the certainty factor of PocketSphinx, for the classification experiments described below.

4.2 Acoustic Features

To produce acoustic features, we leveraged the audio feature extractor openSMILE [13]. We used the baseline acoustic feature sets from the Interspeech (IS) Speaker State Challenge 2011 [5]. The IS 2011 configuration consists of 60 low-level descriptors, providing 4,368 features in total. All audio files were downsampled from 44.1 kHz to 16 kHz sample rate in order to fit the openSMILE configuration.

The used parameters to produce the baseline and our results are shown in table 5. First we oversampled by 127% using 5 nearest neighbors in combination with the Synthetic Minority Oversampling Technique (SMOTE). The classifier is a support vector machine (SVM) using a linear kernel.

The complexity parameter c was tuned on the development set. Details of tuning of c are shown in figure 1.

4.3 Grammar Features

In order to recognize grammar errors in the transcribed samples, we used LanguageTool (v2.4), a proof-reading framework in Java [14].

We applied grammar rules which identify common grammatical and syntactical mistakes in the German language. Altogether, there were 64 features extracted for every recording.

In addition to the experimental results for text and grammar based classification, we tested the combination of text + acoustic + grammar for further optimization.

4.4 Results

Table 6 shows results for all categories using either acoustic, text, grammar or the combination of all three features. The complexity parameter c of the SVM is tuned on the development set. The optimal value of c is also given in table 6.

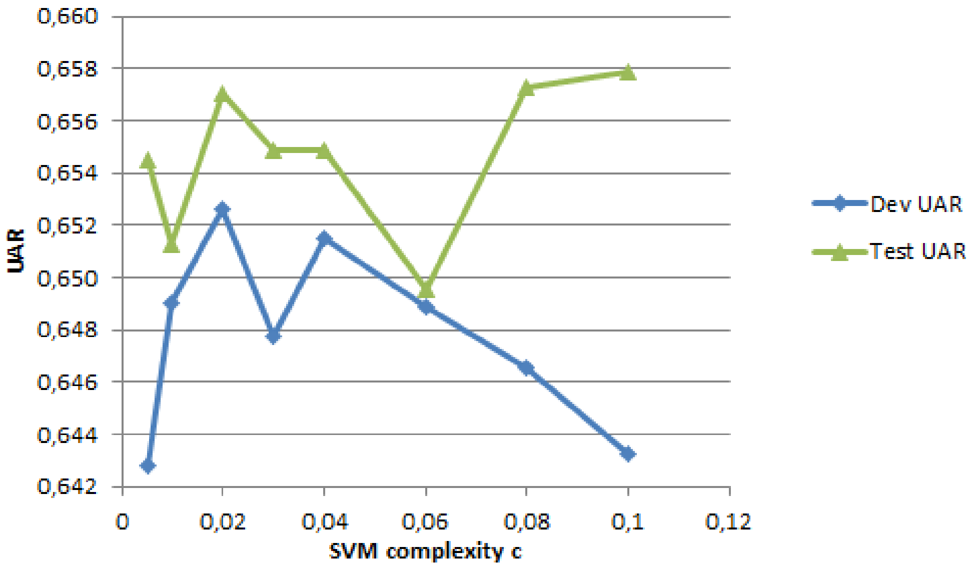

Figure 1 - Tuning the complexity parameter of the SVM

Table 6 - Results for categories with different feature sets

Category	Acoustic			Text			Grammar			Acoustic + Text + Grammar		
	c	UAR dev	UAR test	c	UAR dev	UAR test	c	UAR dev	UAR test	c	UAR dev	UAR test
LN	0.020	63%	66%	0.020	52%	50%	0.040	50%	50%	0.030	63%	66%
LT	0.010	58%	68%	0.060	53%	57%	0.005	50%	50%	0.010	59%	66%
LS	0.010	60%	69%	0.030	53%	50%	0.005	50%	50%	0.010	58%	70%
RT	0.010	64%	70%	0.010	49%	44%	0.005	50%	50%	0.030	65%	71%
RR	0.060	65%	64%	0.005	50%	50%	0.005	50%	50%	0.060	65%	65%
RA	0.010	64%	65%	0.010	48%	47%	0.005	50%	50%	0.020	63%	64%
DQ	0.030	70%	72%	0.010	61%	71%	0.100	55%	50%	0.005	72%	79%
DP	0.080	61%	63%	0.080	69%	61%	0.040	58%	50%	0.005	66%	69%
MQ	0.005	62%	58%	0.030	52%	57%	0.040	54%	50%	0.005	54%	69%
MP	0.030	63%	67%	0.010	68%	66%	0.010	50%	50%	0.030	69%	70%
EC	0.020	66%	66%	0.060	59%	57%	0.020	50%	50%	0.020	65%	65%

Results in green cells outperform the UAR of the IS2011 winner [15] (UAR 70.54%)

Results in yellow cells outperform the baseline [5] (UAR 65.9%)

Comparing the acoustic, text and grammar feature sets shows that acoustic feature classifiers outperform the other feature sets for all categories. It is striking that the grammar based classifiers perform very poorly. One reason for the little information gain provided by the grammar features could be that there are few grammar error types recognized by the framework we use. Moreover, it is obvious that the text classification does not work for categories where participants have to read a given text and are not required to utter spontaneous speech. This is outlined by the poor performance for the reading categories LN, LT, LS, RT, RR and RA. However, text feature based classifiers show better results on spontaneous speech categories i.e. DQ, DP. Unexpectedly, the combination of all features does not improve the results in all cases compared to acoustic features only. One reason for the low performance of the combined feature set could be the simple merging of the features without weighting their impact.

Besides, we compared the performance of transcribed text and by ASR recognized text as shown in figure 2. For five categories ASR generated text features perform better than the text features transcribed by hand. One reason for this finding could be that in addition to the actual text recognized by the ASR, the classifier uses the certainty factor of the ASR.

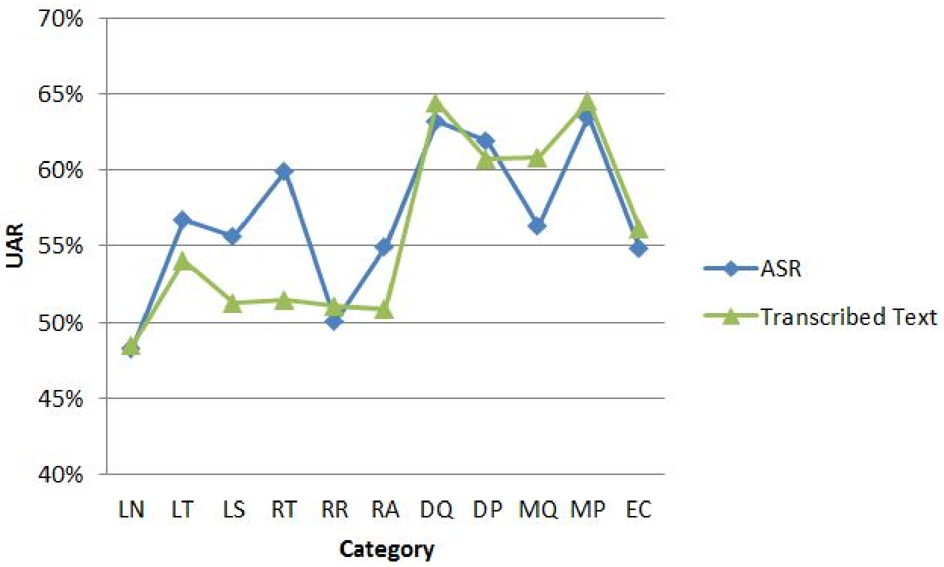

Figure 2 - Transcribed Text vs. ASR classification

5 Conclusion

Our results show that text classification cannot improve previous baseline results from [5], which are based on acoustic features. The main reason is that at least half of the experiments are reading tasks. This means there is hardly any information gain provided by text features.

However, single experiments show that text features can result in additional information gain for experiments of spontaneous speech, e.g. describing pictures (DP) and dialog questions (DQ). Also, we found that using text features produced by ASR does not compromise classification results compared to manual transcriptions.

Moreover, the results point out that the grammar features we applied are not suitable for classifying alcohol intoxication using the given corpus and its experiments. To outperform current baselines the next step could be to use deep learning methods as proposed in [17].

References

[1] Statistisches Bundesamt: Zahl der Verkehrstoten im April 2014 stark gestiegen. Press release. Wiesbaden, Germany. June 2014.

[2] A. Jauch, P. Jaehne, and D. Suendermann: Using Text Classification to Detect Alcohol Intoxication in Speech. In Proc. of the 7th Workshop on Emotion and Computing at the 36th German Conference on Artificial Intelligence, Koblenz, Germany, 2013.

[3] F. Schiel and C. Heinrich: Laying the Foundation for In-Car Alcohol Detection by Speech. In Proc. of the Interspeech, Brighton, UK, 2009.

[4] F. Schiel, C. Heinrich, S. Barfuesser, and T. Gilg: ALC - Alcohol Language Corpus. In Proc. of the LREC, Marrakesh, Morocco, 2008.

[5] B. Schuller, S. Steidl, A. Batliner, F. Schiel, and J. Krajewski: The INTERSPEECH 2011 Speaker State Challenge. Florence, Italy, 2011.

[6] F. Schiel, C. Heinrich, and S. Barfuesser: Alcohol Language Corpus. The first public corpus of alcoholized German speech. Munich, Germany, 2011.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten: The WEKA Data Mining Software: An Update. Orlando, USA. 2009.

[8] C. Chang and C. Lin: LIBSVM: a library for support vector machines. ACM Transactions on Intelligent Systems and Technology (TIST) 2.3 (2011): 27.

[9] D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky: Pocketsphinx: A Free, Real-Time Continuous Speech Recognition System for Hand- Held Devices. In Proc. of the ICASSP, Toulouse, France, 2006.

[10] http://cmusphinx.sourceforge.net/

[11] M. Bisani and H. Ney: Joint-Sequence Models for Grapheme-to-Phoneme Conversion. Speech Communication, Volume 50, Issue 5, Pages 434-451. Amsterdam, The Netherlands, 2008.

[12] Verbmobil Corpus, BAS, Munich, Germany, 1995.

[13] F. Eyben, F. Weninger, F. Gross, and B. Schuller: Recent Developments in openSMILE, the Munich Open-Source Multimedia Feature Extractor. In Proc. ACM Multimedia (MM), Barcelona, Spain, 2013.

[14] M. Miłkowski: Developing an open-source, rule-based proofreading tool. Software-Practice & Experience, Volume 40, Issue 7, New York, USA, 2010.

[15] D. Bone, M. P. Black, M. Li, A. Metallinou, S. Lee, S. S. Narayanan: Intoxicated Speech Detection by Fusion of Speaker Normalized Hierarchical Features and GMM Supervectors. In Proc. of the Interspeech, Florence, Italy, 2011.

[16] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer: "Synthetic minority over-sampling technique“. Journal of Artificial Intelligence Research, vol. 16, pp. 321-357, 2002.

[17] H. Lee, Y. Largman, P. Pham, and A. Y. Ng: Unsupervised feature learning for audio classification using convolutional deep belief networks. In Advances in neural information processing systems (pp. 1096-1104). 2009.