INTERSPEECH 2017
August 20–24, 2017, Stockholm, Sweden

Optimized Time Series Filters for Detecting Laughter and Filler Events

Gábor Gosztolya[1,2]

[1]MTA-SZTE Research Group on Artificial Intelligence, Szeged, Hungary
[2]Department of Informatics, University of Szeged, Hungary
ggabor @ inf.u-szeged.hu

Abstract

Social signal detection, that is, the task of identifying vocalizations like laughter and filler events is a popular task within computational paralinguistics. Recent studies have shown that besides applying state-of-the-art machine learning methods, it is worth making use of the contextual information and adjusting the frame-level scores based on the local neighbourhood. In this study we apply a weighted average time series smoothing filter for laughter and filler event identification, and set the weights using a state-of-the-art optimization method, namely the Covariance Matrix Adaptation Evolution Strategy (CMA-ES). Our results indicate that this is a viable way of improving the Area Under the Curve (AUC) scores: our resulting scores are much better than the accuracy scores of the raw likelihoods produced by Deep Neural Networks trained on three different feature sets, and we also significantly outperform standard time series filters as well as DNNs used for smoothing. Our score achieved on the test set of a public English database containing spontaneous mobile phone conversations is the highest one published so far that was realized by feed-forward techniques.

Index Terms: social signals, laughter events, filler events, time series filter, optimization, evolution strategy

1. Introduction

In speech technology an emerging area is paralinguistic phenomenon detection, which seeks to detect non-linguistic events (laughter, emotions, conflict, etc.) in speech. One task belonging to this area is the detection of social signals, from which, perhaps laughter and filler events (vocalizations like "eh", "er", etc.) are the most important. Many experiments have been performed with the goal of detecting laughter (e.g. [1, 2, 3]), and this task might prove useful in emotion recognition and general man-machine interactions. Apart from laughter, the detection of filler events has also become popular (e.g. [4, 5, 6]). Besides serving to regulate the flow of interactions in discussions, it was also shown that filler events are an important sign of hesitation; hence their detection could prove useful during the automatic detection of various kinds of dementia such as Alzheimer's Disease [5, 7] and Mild Cognitive Impairment [6].

In the tasks of detecting laughter and filler events, classification and evaluation are usually performed at the frame level (although there exist purely segment-based approaches as well; see e.g. [3, 8]). In this approach frames are treated as independent examples, and although it is common to use the feature vectors of the neighbouring frames as well, this provides only minimal contextual information. This is a definite weakness of this approach since these events are typically quite long; the average duration of laughter occurrences was 911 ms in the Hungarian BEA database [9], while in the BMR subset of the ICSI Meeting Recorder Corpus it was 1615 ms with a standard deviation of 1241 ms [10]. Therefore it might be worth making use of the contextual information and adjusting the frame-level scores based on the local neighborhood (see e.g. [11, 12, 13, 14]). Actually, a number of such studies have been published on this. Gupta et al. [11] applied probabilistic time series smoothing; Brueckner et al. [12] trained a second neural network on the output of the first, frame-level one to smooth the resulting scores; Kaya et al. [13] used Gaussian smoothing on the output of frame-level Random Forests; while Gosztolya [14] used the Simple Exponential Smoothing method on the frame-level posterior estimates of DNNs and AdaBoost.MH.

What is common in these studies is that first they trained a frame-level classifier such as Random Forest (RF) or Deep Neural Networks (DNN) to detect the given phenomena, and then, as a second step, they aggregated the neighbouring posterior estimates to get the final scores. Needless to say, the type of smoothing applied was quite different. In this study we compute the weighted mean of the neighbouring DNN output scores as a time series smoothing filter; still, for this type of aggregation, the optimal weight values have to be determined. We shall treat this task as an optimization one in the space of frame-level weights, and maximize the Area Under the Curve (AUC) score for the laughter and filler events. To find the optimal weight values, we apply a state-of-the-art optimization method called the Covariance Matrix Adaptation Evolution Strategy (CMA-ES, [15]). Using the optimal filters found on the development set, we significantly outperform both the unsmoothed ("raw") values and some standard time series filters on the test set of a public English dataset containing laughter and filler events. Furthermore, we also outperform the smoothing approach proposed by Brueckner et al. [12], where a second DNN was used to smooth the likelihoods over time. Overall, the scores we achieved are the highest on this dataset which were realized in a feed-forward way, allowing on-the-fly speech processing.

The structure of this paper is as follows. First we describe the optimization method used. Then, in Section 3, we describe our experimental setup: the database and feature sets used, the way of evaluation, the way we trained our DNNs, the time series filters used for reference, and our optimization approach. Then we present and analyze our test results. Lastly, we draw some conclusions and make some suggestions for future study.

2. Optimization by CMA/ES

We used the Covariance Matrix Adaptation Evolution Strategy (CMA-ES, [15]) to optimize our meta-parameters. Evolution Strategies resemble Genetic Algorithms in that they mimic the evolution of biological populations by selection and recombination, so they are able to "evolve" solutions to real world problems. CMA-ES is a method designed for difficult non-linear non-convex black-box optimization, hence it should be suitable for time series filter optimization for laughter and filler events.

CMA-ES is a second order approach that estimates a covariance matrix within an iterative procedure. This makes the method feasible for badly conditioned, non-smooth (i.e. noisy) or non-continuous problems. It is viewed as a reliable and competitive method for both local and global optimization [16]. It has a further advantage in that it requires little or no meta-parameter setting for optimal performance: it was designed to find optimal or close-to-optimal (strategy) parameters automatically; the aim was to have a well-performing algorithm *as is*. This algorithm has been implemented in several programming languages such as Matlab, Java, C++, Octave and Python. Here, we used the Java implementation with the default settings.

3. Experimental Setup

3.1. The SSPNet Vocalization Corpus

We used the SSPNet Vocalization Corpus [4], which consists of 2763 short audio clips extracted from the English telephone conversations of 120 speakers, containing 2988 laughter and 1158 filler events. Each frame was labeled as one of three classes, namely "laughter", "filler" or "garbage" (meaning both silence and non-filler non-laughter speech). We followed the standard routine of dividing the dataset into training, development and test sets, as published in [17]. The total duration of this dataset is 8 hours and 55 minutes.

3.2. Feature Sets

We applied three different (frame-level) feature sets to train our Deep Neural Networks. The first one was the standard 39-sized MFCC + Δ + $\Delta\Delta$ feature set, while the second one contained 40 raw mel filter bank energies along with energy and their first and second order derivatives (123 values overall). Following the notation of HTK [18], we will refer to the latter feature set as *FBANK*. These two sets were extracted using the HTK tool. The third feature set, referred to as the *ComParE* feature set, was provided for the Interspeech 2013 Computational Paralinguistics Challenge [17], and was extracted with the openSMILE tool [19]. It consisted of the frame-wise 39-long MFCC + Δ + $\Delta\Delta$ feature vector along with voicing probability, HNR, F0 and zero-crossing rate, and their derivatives. To these 47 features their mean and standard derivative in a 9-frame long neighbourhood were added, resulting in a total of 141 features [17].

3.3. Evaluation

As evaluation metrics, we used the method of evaluation which is the de facto standard for laughter detection: we calculated the Area Under Curve (AUC) score for the output likelihood scores of the class of interest. As we now seek to detect two kinds of phenomena (laughter and filler events), we calculated AUC for both social signals; then these AUC values were averaged, giving the Unweighted Average Area Under Curve (UAAUC) score, just as in many previous studies on the SSPNet Vocalization corpus [12, 13, 14, 17, 20].

Since the dataset used had distinct development and test sets, we used the development sets to optimize the time series filter weights; we chose the vector which maximized the AUC on the development set. Then we evaluated the optimal vector on the test set. Since we experimented with two vocalizations (laughter and filler events), which did not necessarily behave in the same way, we set the filter weights independently for the two events, leading to two distinct series of optimization problems.

3.4. Frame-level Classification with DNNs

Before applying a time series filter, first we have to somehow get a likelihood estimate for each class and frame of the utterances. For this, we utilized the classification technique that is now treated as the standard solution for the frame-level phoneme classification (or phoneme posterior estimation) task, namely Deep Neural Networks. We had neurons that used the softmax function in the output layer. Based on the results of our previous experiments (e.g. [14, 21]) and those of preliminary tests, we utilized five hidden layers, each one consisting of 256 rectified linear units. These neurons apply the rectifier activation function $\max(0, x)$ instead of the usual sigmoid one [22]. The main advantage of deep rectifier nets is that they can be trained with the standard backpropagation algorithm, without any tedious pre-training (e.g. [23]). We used our custom DNN implementation, originally developed for phoneme classification. On the TIMIT database, frequently used as a reference dataset for phoneme recognition, our team achieved the lowest phonetic error rate published so far [24].

Frame-level DNN training was done on a sliding window of the neighbouring frame-level feature vectors. We determined the optimal value of neighbours via a grid search: we tested using 1, 5, ..., 65 vectors at once, and chose the one that resulted in the highest AUC score on the development set. The time series smoothing filters were optimized using these frame-level output scores, for which we again used the development set.

3.5. Frame-level Likelihood Aggregation

After obtaining the frame-level likelihood estimates of our classifiers (the "raw" scores), in the next part we will aggregate the values in the local neighbourhood in order to improve the AUC scores. We chose the weighted form of the moving average time series filter; that is, for a filter with a width of $2N + 1$ we define the weight values as $w_{-N}, w_{-N+1}, \ldots, w_N \geq 0$ and $\sum_{i=-N}^{N} w_i = 1$. Afterwards, for the jth frame with the raw likelihood estimate a_j we calculate

$$a'_j = \sum_{i=-N}^{N} w_i a_{j+i}. \qquad (1)$$

(Here we used the simplification that, for an utterance consisting of k frames, $a_j = a_1$ for $\forall j \leq 0$, and $a_j = a_k$ for $\forall j > k$.) We then optimized the w_i weight values.

To test whether the (possible) improvements in the AUC scores come from the actual weight vector and not from the fact that we use some kind of aggregation over time, we also tested two simple approaches. In the first one, we took the unweighted average of the raw likelihood estimates; that is, we had $w_i = \frac{1}{2N+1}$ (*constant* filter). It is quite reasonable to expect that the middle frames are more important than those far away from the central frame; we exploited this in our second basic filter, resembling a triangle (*triangular* filter). As the last method tested for comparison, we trained a DNN on the raw likelihoods. To mirror the setup of the other smoothing filters, the input of the DNNs was only the raw likelihood vector associated with the given class (i.e. laughter or filler events) in the given, $2N + 1$ wide sliding window. These DNNs had three hidden layers with 256 rectifier neurons each, and we used the softmax function in the output layer.

Table 1: *The AUC scores for the laughter and filler events achieved by using the different classification and aggregation methods.*

Feature Set	Filter Type	Development set			Test set		
		Lau.	Fil.	Both	Lau.	Fil.	Both
MFCC	—	93.4	96.0	94.7	90.7	86.8	88.8
	Constant	96.3	97.0	96.7	92.7	87.5	90.1
	Triangular	96.5	97.0	96.8	93.2	87.6	90.4
	DNN	96.6	97.1	96.8	93.8	87.8	90.8
	CMA-ES	96.6	97.1	96.8	**94.4**	**88.0**	**91.2**
FBANK	—	94.6	96.4	95.5	91.2	87.4	89.3
	Constant	97.2	96.8	97.0	93.1	87.7	90.4
	Triangular	97.3	96.9	97.1	93.6	87.7	90.7
	DNN	97.4	96.9	97.2	94.7	**87.9**	**91.3**
	CMA-ES	97.4	96.9	97.1	**95.0**	87.7	**91.3**
ComParE	—	93.8	96.2	95.0	91.8	88.1	89.9
	Constant	97.2	96.7	96.9	94.4	88.4	91.4
	Triangular	97.4	96.7	97.0	95.3	88.5	91.9
	DNN	97.4	96.8	97.1	95.3	88.7	92.0
	CMA-ES	97.5	96.7	97.1	**96.0**	**90.1**	**93.1**
DNN + Prob. time series smoothing [11]		95.1	94.7	94.9	93.3	89.7	91.5
DNN + DNN [12]		98.1	96.5	97.3	94.9	89.9	92.4
ComParE 2013 baseline [17]		86.2	89.0	87.6	82.9	83.6	83.3

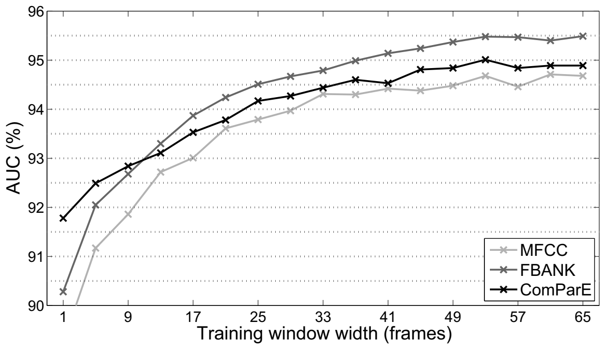

Figure 1: *The AUC scores averaged for the two vocalization types measured on the development set, as a function of the sliding window width used during DNN training.*

3.6. Optimizing the Filter Weights

We represented each time series filter by a vector of the w_i weights. The width of the filters was also determined via a grid search: we tested 9, 17, . . . , 193 frame-wide filters (i.e. 4, 8, . . . , 96 frames on both sides). We supposed that the optimal weights of the neighbouring frames are not completely independent of each other, so we only stored one weight for every four frames, while we linearly interpolated the weight values for the intermediate frames. This approach resulted in a more compact weight vector (e.g. a 193 frames wide filter was represented by only 49 values overall), which is likely to be easier to optimize. Since we had input likelihoods got by using three feature sets, and we optimized the filters for two types of vocalizations (i.e. laughter and filler events), we had 144 optimization problems overall.

To keep the filter hypotheses on the same scale, we first rejected the vectors where the sum of the weight values was outside the range $[0.8, 1.2]$. The CMA-ES optimization method allows us to set the vector where the search process is initiated; we used the appropriate flat filter for this purpose.

4. Results

4.1. Baseline Scores

Figure 1 shows the AUC scores we obtained on the dev set, averaged out for the two types of vocalizations (i.e. laughter and filler events), without using time series filters. It is clear that training the DNNs on a wider sliding window of input frame-level feature vectors is beneficial for all three feature sets, although the actual mean AUC scores and the optimal number of frames used vary. In the following we used the models trained on 61, 65 and 53 consecutive frames, MFCC, FBANK and ComParE feature sets, respectively.

Table 1 lists the output AUC and UAAUC scores we got for the three feature sets and the time series filter approaches. The first thing to notice is that the raw scores (indicated by the "—" filter type) are quite competitive, compared to the ComParE baseline, which were not smoothed over time either. The reason for this is probably that we used DNNs instead of SVM applied by Schuller et al. [17], and that we exploited the feature vectors of the neighbouring frames as well.

As regards the feature sets, we can see that the models trained on MFCCs performed the worst among the three sets tested. Since it is well known (see e.g. [25]) that DNNs work better on more primitive features like mel filter bank energies, it is not surprising that we got higher AUC values by using the FBANK feature set. Furthermore, we can also see that utilizing the ComParE feature set brought the highest AUC values for both vocalization types on the test set; hence it seems to be the most robust one among the three tested sets.

4.2. Basic Filters

Upon examining the two basic smoothing approaches used for reference (filters "constant" and "triangular"), we can see that applying these approaches alone brings a surprisingly large improvement over the raw likelihood scores. This indicates that just by utilizing a smoothing filter of this width (which is usually over a second long) we can noticeably improve the AUC values of the likelihood estimates. Among the two, triangular

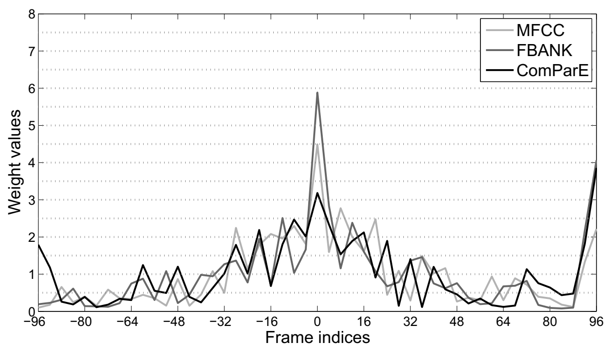

Figure 2: *The optimal filters found by CMA-ES for laughter events.*

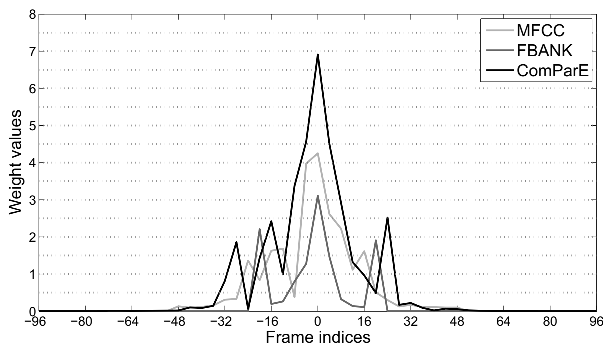

Figure 3: *The optimal filters found by CMA-ES for filler events.*

filters seem to consistently work better for both phenomena that we sought to classify, which confirms our expectation that the frame values around the middle have a much greater importance than those at the sides.

4.3. DNNs and CMA-ES

Utilizing DNNs as time series smoothing filters led to further improvements over the basic filter types: we experienced an improvement on the test set in the range $0.2 - 1.1\%$ over the values got by triangular filters, depending on the vocalization type and feature set. Surprisingly, though, on the development set the difference was usually much smaller.

By using the CMA-ES optimization method, for the FBANK feature set we got quite similar results on the test set; on the MFCC and ComParE feature sets, however, we significantly outperformed the DNN filter (and, with little surprise, the other kinds of filters as well). This, in our opinion, confirms that simply calculating the weighted mean of the frame-wise likelihood scores is a viable way of improving Area-Under-Curve values, and optimizing the weights of the filter via the CMA-ES method works well in practice. (Of course, other optimization methods such as Particle Swarm Optimization [26] or Bacterial Foraging [27] might also be used for this kind of optimization.) A further advantage of our approach is that it is computationally very cheap, especially compared to using Deep Neural Networks or bidirectional Recurrent Neural Networks; while it still allows feed-forward utterance processing.

5. The Time Series Filters Found

Figures 2 and 3 show the time series smoothing filters found by using CMA-ES for the laughter and filler events, respectively. The weight values were scaled up to have a mean of one for better readability (i.e. a weight value of 1 means average importance for the given frame). The large straight sections are due to the linear interpolation of the intermediate frames. It can be seen that the filters are not really smooth themselves, which is probably due to the optimization technique used. Despite this, the filters obtained on the three different feature sets are quite similar to each other for both types of vocalizations.

The filters found for the laughter events have slightly higher weight values around the central frame than those further away (although this tendency is spoiled by the noise present in the weight vectors, which is probably due to the random population initialization of GA). However, what is quite interesting is that the last weight values are quite high, being 2-4 times the average weight. (This, to a lesser extent, also holds for the first frame of the filter found for the ComParE feature set.) For an explanation of this phenomenon, recall that our DNNs were trained using the feature vectors of 26-32 neighbouring frames on both sides. This means that the posterior estimate provided by a DNN for the first frame in the smoothing filter already includes some information about the preceding frames, and using the likelihood estimate of the last frame we can "peek" into the following frames. This makes the first and last frames in the averaged filter more important than the inner ones, while the values of the inner frames are redundant to some extent.

Examining the filler events, we can see that they are also quite similar to each other. Furthermore, the middle frames seem to be very important, having a relative importance of about 3-7 times that of an average frame. Another clear difference among the two phenomena is that much wider filters were needed for the two vocalization types for optimal performance on the development set: for laughter events, all the three optimal filters had the maximal length, being almost two seconds long, while for filler events this value was only between 41 and 129. This is probably due to the difference in the typical length of the two vocalization types: in this database, laughter events have a mean duration of 942 milliseconds, while filler events are 502 ms long on average.

6. Conclusions

In this study, we investigated the task of laughter and filler detection. As was shown earlier, after performing some frame-level posterior estimation step via some machine learning method, it is worth smoothing the output likelihood scores of the consecutive frames. This was why we applied a weighted averaging time series smoothing filter. To set the weights in the filter, we applied the state-of-the-art optimization method of CMA-ES, using the development sets of a public English dataset. Our AUC scores got on the test set significantly outperformed both the unsmoothed likelihood values and standard time series filters of the same size, while we also got better results than by just utilizing DNNs. Overall, we report the highest average AUC score on the test set achieved by a feed-forward technique. It would be interesting to see the amount of language independence of the filters found; this, however, is the subject of future works.

7. Acknowledgements

The Titan X graphics card used for this study was donated by the NVIDIA Corporation.

8. References

[1] L. S. Kennedy and D. P. W. Ellis, “Laughter detection in meetings,” in *Proceedings of the NIST Meeting Recognition Workshop at ICASSP*, Montreal, Canada, 2004, pp. 118–121.

[2] K. P. Truong and D. A. van Leeuwen, “Automatic detection of laughter,” in *Proceedings of Interspeech*, Lisbon, Portugal, 2005, pp. 485–488.

[3] T. Neuberger, A. Beke, and M. Gósy, “Acoustic analysis and automatic detection of laughter in Hungarian spontaneous speech,” in *Proceedings of ISSP*, 2014, pp. 281–284.

[4] H. Salamin, A. Polychroniou, and A. Vinciarelli, “Automatic detection of laughter and fillers in spontaneous mobile phone conversations,” in *Proceedings of SMC*, 2013, pp. 4282–4287.

[5] I. Hoffmann, D. Németh, C. Dye, M. Pákáski, T. Irinyi, and J. Kálmán, “Temporal parameters of spontaneous speech in Alzheimer’s disease,” *International Journal of Speech-Language Pathology*, vol. 12, no. 1, pp. 29–34, 2010.

[6] L. Tóth, G. Gosztolya, V. Vincze, I. Hoffmann, G. Szatlóczki, E. Biró, F. Zsura, M. Pákáski, and J. Kálmán, “Automatic detection of Mild Cognitive Impairment from spontaneous speech using ASR,” in *Proceedings of Interspeech*, Dresden, Germany, Sep 2015, pp. 2694–2698.

[7] J. Weiner, C. Herff, and T. Schultz, “Speech-based detection of Alzheimer’s disease in conversational German,” in *Proceedings of Interspeech*, San Francisco, CA, USA, Sep 2016, pp. 1938–1942.

[8] N. Campbell, H. Kashioka, and R. Ohara, “No laughing matter,” in *Proceedings of Interspeech*, Lisbon, Portugal, 2005, pp. 465–468.

[9] T. Neuberger and A. Beke, “Automatic laughter detection in Hungarian spontaneous speech using GMM/ANN hybrid method,” in *Proceedings of SJUSK Conference on Contemporary Speech Habits*, 2013, pp. 1–13.

[10] M. T. Knox and N. Mirghafori, “Automatic laughter detection using neural networks,” in *Proceedings of Interspeech*, Antwerp, Belgium, 2007, pp. 2973–2976.

[11] R. Gupta, K. Audhkhasi, S. Lee, and S. S. Narayanan, “Speech paralinguistic event detection using probabilistic time-series smoothing and masking,” in *Proceedings of InterSpeech*, 2013, pp. 173–177.

[12] R. Brueckner and B. Schuller, “Hierarchical neural networks and enhanced class posteriors for social signal classification,” in *Proceedings of ASRU*, 2013, pp. 362–367.

[13] H. Kaya, A. Ercetin, A. Salah, and S. Gürgen, “Random forests for laughter detection,” in *Proceedings of WASSS*, 2013.

[14] G. Gosztolya, “On evaluation metrics for social signal detection,” in *Proceedings of Interspeech*, Dresden, Germany, Sep 2015, pp. 2504–2508.

[15] N. Hansen and A. Ostermeier, “Completely derandomized self-adaptation in evolution strategies,” *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[16] N. Hansen and S. Kern, “Evaluating the CMA evolution strategy on multimodal test functions,” in *Parallel Problem Solving from Nature PPSN VIII*, ser. LNCS, X. Yao *et al.*, Eds., vol. 3242. Springer, 2004, pp. 282–291.

[17] B. Schuller, S. Steidl, A. Batliner, A. Vinciarelli, K. Scherer, F. Ringeval, M. Chetouani, F. Weninger, F. Eyben, E. Marchi, H. Salamin, A. Polychroniou, F. Valente, and S. Kim, “The Interspeech 2013 Computational Paralinguistics Challenge: Social signals, Conflict, Emotion, Autism,” in *Proceedings of Interspeech*, 2013.

[18] S. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. Woodland, *The HTK Book*. Cambridge, UK: Cambridge University Engineering Department, 2006.

[19] F. Eyben, M. Wöllmer, and B. Schuller, “Opensmile: The Munich versatile and fast open-source audio feature extractor,” in *Proceedings of ACM Multimedia*, 2010, pp. 1459–1462.

[20] R. Brueckner and B. Schuller, “Social signal classification using deep BLSTM recurrent neural networks,” in *Proceedings of ICASSP*, 2014, pp. 4856–4860.

[21] G. Gosztolya, “Detecting laughter and filler events by time series smoothing with genetic algorithms,” in *Proceedings of SPECOM*, Budapest, Hungary, Aug 2016, pp. 232–239.

[22] X. Glorot, A. Bordes, and Y. Bengio, “Deep sparse rectifier networks,” in *Proc. AISTATS*, 2011, pp. 315–323.

[23] T. Grósz and L. Tóth, “A comparison of Deep Neural Network training methods for Large Vocabulary Speech Recognition,” in *Proceedings of TSD*, Pilsen, Czech Republic, 2013, pp. 36–43.

[24] L. Tóth, “Phone recognition with hierarchical Convolutional Deep Maxout Networks,” *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2015, no. 25, pp. 1–13, 2015.

[25] A. Mohamed, G. E. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” *IEEE Trans. ASLP*, vol. 20, no. 1, pp. 14–22, 2012.

[26] J. Kennedy and R. Eberhart, “Particle Swarm Optimization,” in *Proceedings of ICNN*, Perth, Australia, 1991, pp. 1942–1948.

[27] K. M. Passino, “Biomimicry of bacterial foraging for distributed optimization and control,” *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.