Probabilistic speech feature extraction with context-sensitive Bottleneck neural networks

Martin Wöllmer*, Björn Schuller

Institute for Human-Machine Communication, Technische Universität München, Theresienstr. 90, 80333 München, Germany

1. Introduction

Considering the unsatisfying word accuracies that occur whenever today's automatic speech recognition (ASR) systems are faced with 'unfriendly' scenarios such as conversational and disfluent speaking styles, emotional coloring of speech, or distortions caused by noise, the need for novel concepts that go beyond main stream ASR techniques becomes clear [1,2]. Since systems that are exclusively based on conventional generative Hidden Markov Models (HMM) appear to be limited in their reachable modeling power and recognition rates, the combination of Markov modeling and discriminative techniques such as neural networks has emerged as a promising method to cope with challenging ASR tasks. Hence, Tandem front-ends that apply multi-layer perceptrons (MLP) or recurrent neural networks (RNN) to generate probabilistic features for HMM processing are increasingly used in modern ASR systems [3–6].

Such Tandem systems apply neural networks to map from standard low-level speech features like Mel-Frequency Cepstral Coefficients (MFCC) or Perceptual Linear Prediction (PLP) features to phoneme or phoneme state posteriors which in turn can be used as features within an HMM framework. Usually, the quality of those probabilistic features heavily depends on the phoneme recognition accuracy of the underlying neural network. As phoneme recognition is known to profit from context modeling, an obvious strategy to consider contextual information is to use a stacked sequence of past and future vectors as input for an MLP that generates phoneme predictions [7]. However, extensive experiments in [8] have shown that flexible context modeling *within* the neural network leads to better phoneme recognition results than processing fixed-length feature vector sequences. Bidirectional Long-Short Term Memory (BLSTM) recurrent neural networks based on the concept introduced in [9] and refined in [10,8] were shown to outperform comparable context-sensitive phoneme recognition architectures such as MLPs, RNNs, or triphone HMMs, as they are able to model a self-learned amount of context via recurrently connected memory blocks. Thus, it seems promising to exploit the concept of BLSTM in Tandem ASR systems.

First attempts to use BLSTM networks for speech recognition tasks can be found in the area of keyword spotting [11–13]. In [14] it was shown that also continuous speech recognition performance can be enhanced when using a discrete feature, that indicates the current phoneme identity determined by a BLSTM network, in addition to MFCC features. Further performance gains could be demonstrated in [15] by applying a multi-stream HMM framework that models MFCC features and the discrete BLSTM phoneme estimate as two independent data streams. An enhanced BLSTM topology for multi-stream BLSTM-HMM modeling was presented in [6], leading to further ASR improvements.

An interesting alternative to using the logarithmized and decorrelated activations of the *output* layer of recurrent neural networks (RNN) or multi-layer perceptrons (MLP) as probabilistic

* Corresponding author. Tel.: +49 89 289 28550; fax: +49 89 289 28535.
E-mail address: woellmer@tum.de (M. Wöllmer).

features is the extraction of activations in a narrow *hidden* layer within the network as the so-called 'Bottleneck' (BN) features [16]. This implies the advantage that the size of the feature space can be chosen by defining the size of the network's Bottleneck layer which makes the dimension of the feature vectors independent of the number of network training targets. The linear outputs of the Bottleneck layer are usually well decorrelated and do not have to be logarithmized.

In this paper, we present and optimize a novel approach towards BLSTM feature generation for Tandem ASR. First, we replace the discrete phoneme prediction feature used in [6] by the continuous logarithmized vector of BLSTM output activations and merge it with low-level MFCC features. By that we obtain extended context-sensitive Tandem feature vectors that lead to improved results when evaluated on the COSINE [17] and the Buckeye [18] corpora. Then, we show how ASR performance can be further enhanced by uniting the concepts of BLSTM and Bottleneck feature extraction.

The structure of our paper is as follows: First, in Section 2, we explain the BLSTM technique and provide an overview over the previously proposed BLSTM-based ASR systems. Next, we introduce our BN-BLSTM front-end (Section 3) and the used spontaneous speech corpora (Section 4). Finally, in Section 5, we present our experiments and results.

2. Background

2.1. Bidirectional Long Short-Term Memory RNNs

Even though the recurrent connections in RNNs allow to model contextual information, which makes them a more effective sequence labeling framework than, for example, MLPs, it is known that the range of context that standard RNNs can access is limited [19]. The reason for this is that the influence of a certain input on the hidden and output layer of the network either blows up or decays exponentially over time while cycling around the recurrent connections of the network. In literature, this problem is referred to as the so-called *vanishing gradient problem*. The effect of this decaying sensitivity is that RNNs have difficulties in learning temporal dependencies for which relevant inputs and targets are separated by more than ten time steps [19], i.e., the network cannot *remember* the previous inputs over longer time spans so that it is hardly possible to model input-target dependencies that are not synchronous. One of the most effective techniques to overcome this problem is the Long Short-Term Memory architecture [9], which is able to store information in linear memory cells over a longer period of time.

An LSTM hidden layer is composed of multiple recurrently connected subnets which will be referred to as *memory blocks* in the following. Every memory block consists of self-connected *memory cells* and three multiplicative *gate* units (input, output, and forget gates). Since these gates allow for write, read, and reset operations within a memory block, an LSTM block can be interpreted as (differentiable) memory chip in a digital computer. Fig. 1 contains an illustration of the architecture of a memory block comprising one memory cell.

If α_t^{in} denotes the activation of the input gate at time t *before* the activation function f_g has been applied and β_t^{in} represents the activation *after* application of the activation function, the input gate activations (forward pass) can be written as

$$\alpha_t^{\text{in}} = \sum_{i=1}^{I} \eta^{i,\text{in}} x_t^i + \sum_{h=1}^{H} \eta^{h,\text{in}} \beta_{t-1}^h + \sum_{c=1}^{C} \eta^{c,\text{in}} s_{t-1}^c \tag{1}$$

and

$$\beta_t^{\text{in}} = f_g(\alpha_t^{\text{in}}), \tag{2}$$

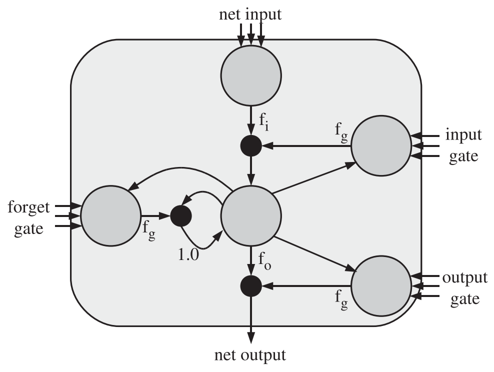

Fig. 1. LSTM memory block consisting of one memory cell: the input, output, and forget gates collect activations from inside and outside the block which control the cell through multiplicative units (depicted as small circles); input, output, and forget gate scale input, output, and internal state; f_i, f_g, and f_o denote activation functions; the recurrent connection of fixed weight 1.0 maintains the internal state.

respectively. The variable η^{ij} corresponds to the weight of the connection from unit i to unit j while 'in', 'for', and 'out' refer to input gate, forget gate, and output gate, respectively. Indices i, h, and c count the inputs x_t^i, the cell outputs from other blocks in the hidden layer, and the memory cells, respectively, while I, H, and C are the number of inputs, the number of cells in the hidden layer, and the number of memory cells. Finally, s_t^c corresponds to the *state* of a cell c at time t, meaning the activation of the linear cell unit.

Similarly, the activation of the forget gates before and after applying f_g can be calculated as follows:

$$\alpha_t^{\text{for}} = \sum_{i=1}^{I} \eta^{i,\text{for}} x_t^i + \sum_{h=1}^{H} \eta^{h,\text{for}} \beta_{t-1}^h + \sum_{c=1}^{C} \eta^{c,\text{for}} s_{t-1}^c \tag{3}$$

$$\beta_t^{\text{for}} = f_g(\alpha_t^{\text{for}}). \tag{4}$$

The memory cell value α_t^c is a weighted sum of inputs at time t and hidden unit activations at time $t-1$:

$$\alpha_t^c = \sum_{i=1}^{I} \eta^{i,c} x_t^i + \sum_{h=1}^{H} \eta^{h,c} \beta_{t-1}^h. \tag{5}$$

To determine the current state of a cell c, we scale the previous state by the activation of the forget gate and the input $f_i(\alpha_t^c)$ by the activation of the input gate:

$$s_t^c = \beta_t^{\text{for}} s_{t-1}^c + \beta_t^{\text{in}} f_i(\alpha_t^c). \tag{6}$$

The computation of the output gate activations follows the same principle as the calculation of the input and forget gate activations, however, this time we consider the *current* state s_t^c, rather than the state from the previous time step:

$$\alpha_t^{\text{out}} = \sum_{i=1}^{I} \eta^{i,\text{out}} x_t^i + \sum_{h=1}^{H} \eta^{h,\text{out}} \beta_{t-1}^h + \sum_{c=1}^{C} \eta^{c,\text{out}} s_t^c \tag{7}$$

$$\beta_t^{\text{out}} = f_g(\alpha_t^{\text{out}}). \tag{8}$$

Finally, the memory cell output is determined as

$$\beta_t^c = \beta_t^{\text{out}} f_o(s_t^c). \tag{9}$$

The overall effect of the gate units is that the LSTM memory cells can store and access information over long periods of time and thus avoid the vanishing gradient problem. For instance, as long as the input gate remains closed (corresponding to an input gate activation close to zero), the activation of the cell will not be overwritten by new inputs and can therefore be made available to the net much later in the sequence by opening the output gate.

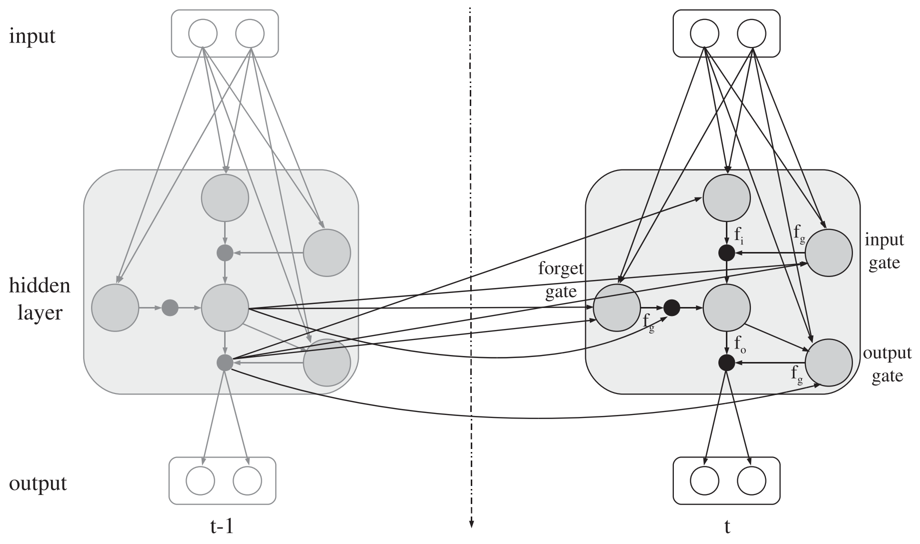

Fig. 2. Connections in an LSTM network consisting of two input nodes, one memory cell with one memory block, and two output nodes.

Fig. 2 provides an overview over the connections in an 'unrolled' LSTM network for time steps $t-1$ and t. For the sake of simplicity, this network only contains small input and output layers (two nodes each) and just one memory block with one cell. Note that the initial version of the LSTM architecture contained only input and output gates. Forget gates were added later [10] in order to allow the memory cells to reset themselves whenever the network needs to *forget* past inputs. In this paper, we exclusively consider the enhanced LSTM version including forget gates.

Another shortcoming of standard RNNs is that they have access to past but not to future context. This can be overcome by using *bidirectional* recurrent neural networks (BRNN) [20], where two separate recurrent hidden layers scan the input sequences in opposite directions. The two hidden layers are connected to the same output layer, which therefore has access to context information in both directions. The amount of context information that the network actually uses is learned during training, and does not have to be specified beforehand. In this paper we use a combination of the principle of bidirectional networks and the LSTM technique (i.e., bidirectional LSTM). Of course the usage of bidirectional context implies a short look-ahead buffer, meaning that recognition cannot be performed truly on-line. However, for many speech recognition tasks it is sufficient to obtain an output, e.g., at the end of an utterance, so that both the forward and the backward context can be used during decoding.

In recent years, the LSTM technique has been successfully applied for a variety of pattern recognition tasks, including phoneme classification [8], emotion recognition [13,21], handwriting recognition [22,23], and driver distraction detection [24].

2.2. BLSTM modeling for continuous speech recognition

Previous approaches towards continuous speech recognition exploiting BLSTM context-modeling concentrated on appending a discrete BLSTM feature to the (continuous) acoustic feature vector. This additional feature b_t encodes the framewise phoneme prediction generated via a BLSTM network, i.e., it corresponds to the index of the most active output activation which in turn corresponds to a certain phoneme at a given time step (see [14] for formulas). Applying the resulting extended feature vector $y_t = [x_t; b_t]$, that contains MFCC features x_t and the BLSTM phoneme estimate b_t, was shown to boost recognition performance of keyword detectors [12] and continuous ASR systems [14]. Further performance gains could be obtained by employing a multi-stream HMM to model x_t and b_t as two independent data streams [15] which allows to introduce different stream weights for low-level acoustic features and BLSTM phoneme predictions. Modeling long-range feature-level context via bidirectional Long Short-Term Memory could outperform simple feature frame stacking, as it is done in conventional Tandem ASR systems [6]. Thus, the application of BLSTM appears to be a promising method to generate enhanced Tandem features for speech recognition.

3. System overview

3.1. BLSTM feature extraction

The flowchart in Fig. 3 provides an overview over our ASR system employing BLSTM feature extraction. Cepstral mean and variance normalized MFCC features, including coefficients 1–12, logarithmized energy, as well as first and second order temporal derivatives, build a 39-dimensional feature vector which serves as input for our BLSTM network. We use the common framerate of 10 ms and a window size of 25 ms. The BLSTM network is trained on framewise phoneme targets and thus generates a vector of output activations whose entries correspond to estimated phoneme posteriors. Since the network uses a 'softmax' activation function for the output layer, the output activations are approximately gaussianized via mapping to the logarithmic domain. The number of BLSTM features per time frame corresponds to the number of distinct phoneme targets (41 for the COSINE experiment, see Section 5). Merging BLSTM features and the original normalized MFCC features into one large feature vector, we obtain 80 Tandem features that are processed via principal component analysis (PCA) in order to decorrelate and compress the feature space by using only 40 principal components as features. The final feature vector is forwarded to an HMM-based ASR system generating the word hypothesis.

Figs. 4(a)–5(b) show the processing steps for an example speech sequence. The MFCC feature vectors are hardly correlated and approximately follow a Gaussian distribution (Fig. 4(a)). Due to the softmax activation function generating the outputs of the BLSTM phoneme predictor, the network tends to produce sharp spikes that indicate the presence of a particular phoneme at a particular timestep (Fig. 4(b)). To gaussianize the outputs, we apply the logarithm (Fig. 5(a)). Finally, we merge BLSTM and MFCC

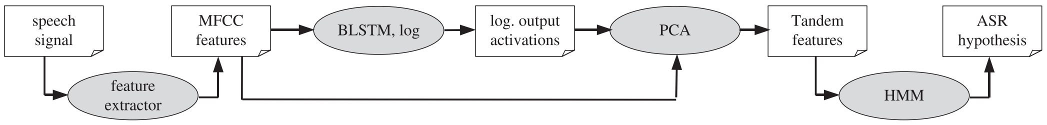

Fig. 3. Tandem BLSTM front-end incorporated into an HMM-based ASR system.

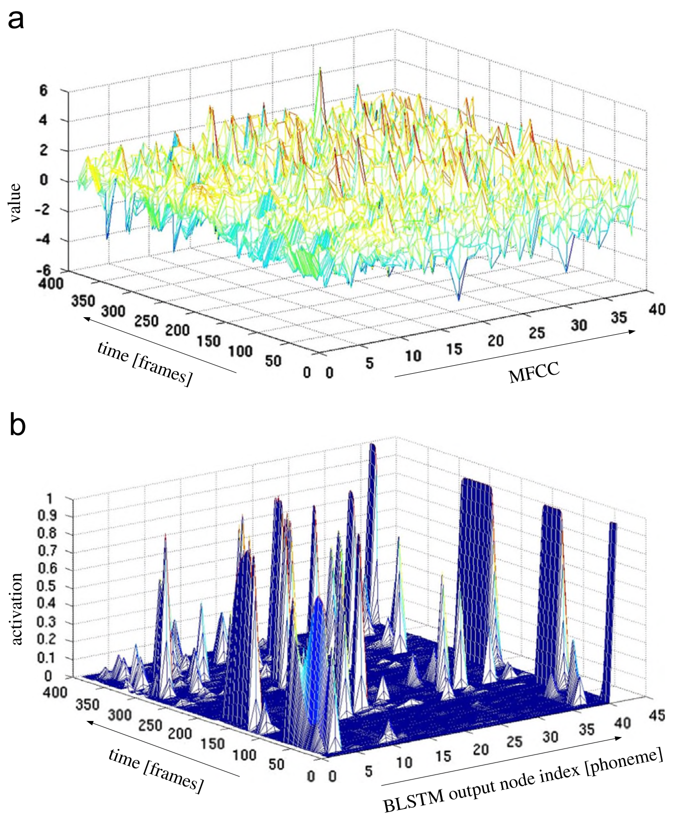

Fig. 4. Input and output of a BLSTM phoneme predictor processing a speech sequence. (a) Normalized MFCC features (including first and second order derivatives) over time. (b) Raw output activations of the BLSTM network.

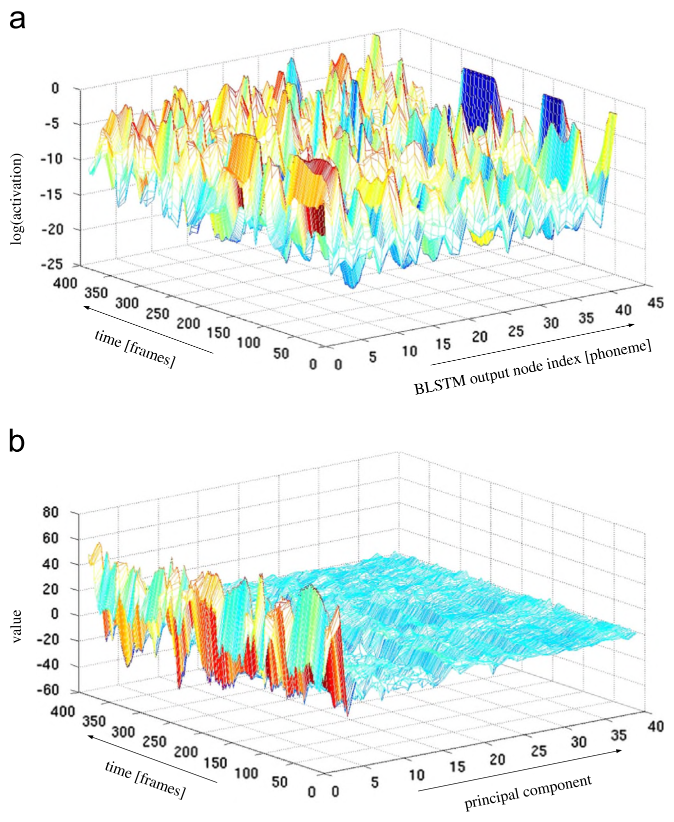

Fig. 5. Post-processing of BLSTM features: Mapping to the logarithmic domain (a), subsequent concatenation of MFCC features, and PCA transformation (b). Only the principal components corresponding to the 40 largest eigenvalues are shown.

features and decorrelate the resulting feature vector sequence via PCA (Fig. 5(b)).

3.2. Bottleneck-BLSTM front-end

The Bottleneck-BLSTM feature extractor investigated in this paper can be seen as a combination of bidirectional LSTM modeling for improved context-sensitive Tandem feature generation and Bottleneck front-ends. The Bottleneck principle allows to generate Tandem feature vectors of arbitrary size by using the activations of the hidden (Bottleneck) layer as features – rather than the logarithmized output activations corresponding to the estimated phoneme or phoneme state posteriors. Since we focus on *bidirectional* processing, we have two Bottleneck layers: one within the network processing the speech sequence in forward direction and the other within the network for backward processing. Fig. 6 shows the system flowchart of our ASR system based on BN-BLSTM features. Again, 39 cepstral mean and variance normalized MFCC features are extracted from the speech signal. These features serve as input for a BN-BLSTM network that is trained on framewise phoneme targets. During feature extraction, the activations of the output layer are ignored; only the activations of the forward and backward Bottleneck layers are processed (i.e., the memory block outputs of the Bottleneck layers). Together with the original MFCC features, the forward and backward Bottleneck layer activations are concatenated to one feature vector which is then decorrelated by Principal Component Analysis (PCA).

Fig. 7 illustrates the detailed structure of the applied Bottleneck-BLSTM front-end. The input activations of the network correspond to the normalized MFCC features. Three hidden LSTM layers are used per input direction. Best performance could be obtained when using a hidden layer of size 78 (two times the number of MFCC features) as first hidden LSTM layer, a second hidden layer of size 128, and a comparably narrow third hidden layer, representing the Bottleneck (size 20–80). Network parameters such as learning rate, momentum, and initialization weights have not been tuned but were set according to past experiments on LSTM-based phoneme prediction, e.g., as in [6]. In all our experiments, we use identical topologies for forward and backward Bottleneck layers. The connections between the Bottleneck layers and the output layer are depicted in gray, indicating that the activations of the output layer (o_t) are only used during network training and not during BN-BLSTM feature extraction. To obtain the final decorrelated feature vectors, PCA is applied on the joint feature vectors consisting of forward and backward Bottleneck layer activations and MFCC features x_t.

4. Spontaneous speech corpora

We optimized and evaluated our BLSTM feature extraction scheme on the 'COnversational Speech In Noisy Environments'

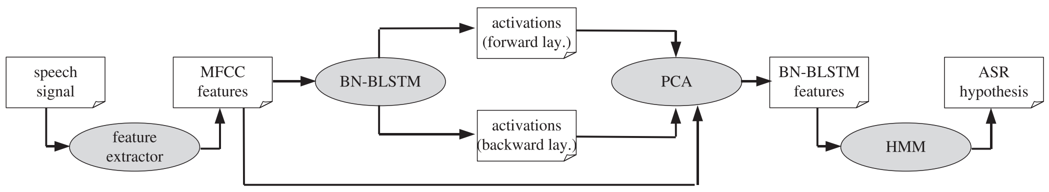

Fig. 6. Bottleneck-BLSTM front-end incorporated into an HMM-based ASR system.

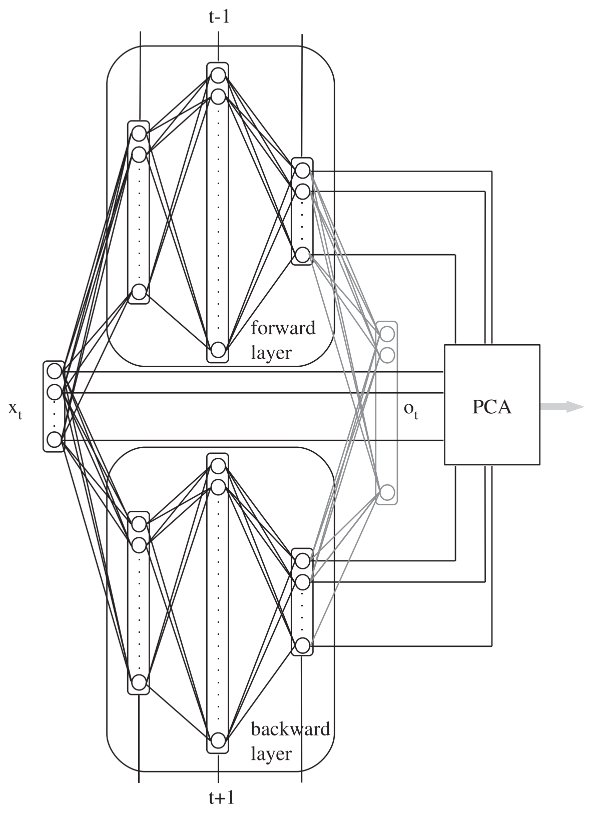

Fig. 7. Architecture of the Bottleneck-BLSTM front-end.

(COSINE) corpus [17] which is a relatively new database containing multi-party conversations recorded in real world environments. The COSINE corpus has also been used in [14,15,6] which allows us to compare the proposed front-end to the previously introduced concepts for BLSTM-based feature-level context modeling in continuous ASR.

The COSINE recordings were captured on a wearable recording system so that the speakers were able to walk around during recording. Since the participants were asked to speak about anything they liked and to walk to various noisy locations, the corpus consists of natural, spontaneous, and highly disfluent speaking styles partly masked by indoor and outdoor noise sources such as crowds, vehicles, and wind. The recordings were captured with multiple microphones simultaneously, however, to match most application scenarios, we focused on speech recorded by a close-talking microphone. We used all ten transcribed sessions, containing 11.40 h of pairwise English conversations and group discussions (37 speakers). For our experiments, we applied the recommended test set (sessions 3 and 10) which comprises 1.81 h of speech. Sessions 1 and 8 were used as validation set (2.72 h of speech) and the remaining six sessions made up the training set. The vocabulary size of the COSINE corpus is 4.8k.

To verify whether word accuracy improvements obtained via BLSTM features can also be observed for other spontaneous speech scenarios, experiments were repeated applying the Buckeye corpus [18] (without further optimizations). The Buckeye corpus contains recordings of interviews with 40 subjects, who were told that they were in a linguistic study on how people express their opinions. The corpus has been used for a variety of phonetic studies as well as for ASR experiments [25]. Similar to the COSINE database, the contained speech is highly spontaneous. The 255 recording sessions, each of which is approximately 10 min long, were subdivided into turns by cutting whenever a subject's speech was interrupted by the interviewer, or once a silence segment of more than 0.5 s length occurred. We used the same speaker independent training, validation, and test sets as defined in [25]. The lengths of the three sets are 20.7 h, 2.4 h, and 2.6 h, and the vocabulary size is 9.1k.

5. Experiments and results

5.1. Tandem BLSTM front-end

At first, four different variants of our proposed Tandem BLSTM-HMM recognizer (see Section 3.1) were trained and evaluated on the COSINE corpus. The underlying BLSTM network was the same as employed for generating the discrete phoneme prediction feature in [6], i.e., the network consisted of three hidden layers per input direction (size of 78, 128, and 80) and each LSTM memory block contained one memory cell. We trained the network on the standard Carnegie Mellon University (CMU) set of 39 different English phonemes with additional targets for *silence* and *short pause*, using backpropagation through time [26] and a learning rate of 10^{-5}. As a common means to improve generalization for RNNs, we added zero mean Gaussian noise with standard deviation 0.6 to the inputs during training. Prior to training, all weights were randomly initialized in the range from -0.1 to 0.1. Input and output gates used tanh activation functions, while the forget gates had logistic activation functions. Training was aborted as soon as no improvement on the COSINE validation set could be observed for at least 50 epochs. Finally, we chose the network that achieved the best framewise phoneme error rate on the validation set.

Initially, we used only the first 40 principal components of the PCA-processed Tandem feature vector as input for the HMM recognizer, i.e., the principal components corresponding to the 40 largest eigenvalues. Hence, the HMM system was based on the same number of features as the previously proposed BLSTM-based recognizers [14,15,6]. In conformance with [6], the HMM back-end consisted of left-to-right HMMs with three emitting states per phoneme and 16 Gaussian mixtures per state. We applied tied-state cross-word triphone models with shared state transition probabilities and a back-off bigram language model, all trained on the training partition of the COSINE corpus.

In Table 1, the results on the COSINE test set are summarized. Exclusively applying the raw output activations as BLSTM features leads to a word accuracy (WA) of 40.76%. A slight improvement can be observed when taking the logarithm of the estimated phoneme posteriors (WA of 41.24%). Decorrelation via PCA further increases the word accuracy to 44.18% for 40 principal components. Finally, the best Tandem BLSTM-HMM performance is observed for a system as shown in Fig. 3, i.e., an HMM processing PCA-transformed feature vectors that contain both the original MFCC features and the logarithmized BLSTM activations (WA of 48.51% for

Table 1
COSINE test set: word accuracies (WA) obtained for Tandem BLSTM-HMM modeling with and without taking the logarithm (log) of the BLSTM output activations, decorrelation via PCA, and including MFCC features in the final feature vector (prior to PCA); results are obtained using only the first 40 principal components.

Model architecture	log	PCA	MFCC	WA (%)
Tandem BLSTM-HMM	✗	✗	✗	40.76
Tandem BLSTM-HMM	✓	✗	✗	41.24
Tandem BLSTM-HMM	✓	✓	✗	44.18
Tandem BLSTM-HMM	✓	✓	✓	**48.51**
multi-stream BLSTM-HMM [6]	–	✗	✓	48.01
multi-stream BLSTM-HMM [15]	–	✗	✓	46.50
discrete BLSTM feature [14]	–	✗	✓	45.04
HMM	–	✗	✓	43.36

40 principal components). This system prevails over the initial [15] and enhanced [6] versions of a multi-stream BLSTM-HMM modeling MFCCs and a discrete BLSTM phoneme prediction feature as two independent data streams. Also a comparable single-stream HMM system modeling the BLSTM prediction as additional discrete feature (WA of 45.04% [14]) as well as a baseline HMM processing only MFCC features (43.36%) is outperformed by our Tandem BLSTM-HMM.

5.2. Bottleneck-BLSTM front-end

For BN-BLSTM feature extraction according to the flowchart in Fig. 6, we evaluated a number of different Bottleneck network architectures. At first, we considered a BLSTM network with a first and third hidden layer of size 128 and a second (Bottleneck) layer of sizes 20, 40, and 80. Best performance could be obtained with a relatively large Bottleneck of size 80. Next, we trained and validated networks such as the one depicted in Fig. 7, i.e., networks consisting of a first and second hidden layer of size 78 and 128, respectively, with the third hidden layer used as Bottleneck – again evaluating sizes 20, 40, and 80. Network training parameters were set exactly as for the Tandem BLSTM front-end (see Section 5.1). Again, we used only the first 40 principal components as final feature vector. The best word accuracy on the COSINE test set was 49.51% and was achieved with a 78-128-80 hidden layer topology, using the activations of the third hidden layer as features. Thus, prior to PCA, the extended BN feature vector is composed of 199 features (80 activations from the forward hidden layer, 80 activations from the backward hidden layer, and 39 MFCC features). Note that the best hidden layer topology for the BN-BLSTM front-end was the same as used for Tandem front-end investigated in Section 5.1.

Fig. 8 shows the effect the number of PCA coefficients used as features has on recognition performance for evaluations on the COSINE test set. When applying Tandem BLSTM features, we observe comparable word accuracies for feature vector dimensionalities between 35 and 45, with two maxima for 37 coefficients (WA of 48.73%) and 40 coefficients (WA of 48.51%). When employing the BN-BLSTM feature extractor, we can see a clear global maximum of WA for 39-dimensional feature vectors (WA of 49.92%). For feature vector sizes larger than 37, the Bottleneck system prevails over the Tandem BLSTM-HMM.

5.3. Effect of LSTM and bidirectional modeling

To investigate the effect of Long Short-Term Memory and bidirectional modeling, we replaced the (BN-)BLSTM networks in Figs. 3 and 6 by unidirectional LSTM networks and bi- or unidirectional RNNs, respectively. Optimizations of the hidden layer topology were carried out for each network type individually. For experiments on the COSINE database, all LSTM, BRNN, and RNN-based front-ends applied the 78-128-80 hidden layer topology which led to the best result for each network type. Prior to using the Tandem and Bottleneck features for continuous ASR, we evaluated the framewise phoneme accuracy (FPA) of the underlying neural network architectures. As can be seen in the second column of Table 2, bidirectional LSTM networks perform notably better than unidirectional LSTM nets and LSTM architectures outperform conventional RNNs.

All word accuracies shown in Table 2 are based on feature vectors of size 39 (except for the results taken from [14,6], which are obtained using 39+1 features, see Section 2.2). The third column of Table 2 shows the word accuracies for systems trained and evaluated on the COSINE corpus. When applying bidirectional processing, front-ends using Bottleneck activations from the third hidden layer outperform Tandem systems processing the logarithmized output activations. For both front-end types, RNN architectures cannot compete with LSTM architectures, which shows the importance of long-range context modeling in challenging spontaneous and disfluent speech scenarios. The Bottleneck BLSTM features (leading to a WA of 49.92%, see Section 5.2) prevail over comparable BLSTM features based on continuous output activations (48.23%), as well as over the multi-stream BLSTM-HMM technique [6] applying combined continuous-discrete modeling of MFCC features and BLSTM phoneme predictions (48.01%).

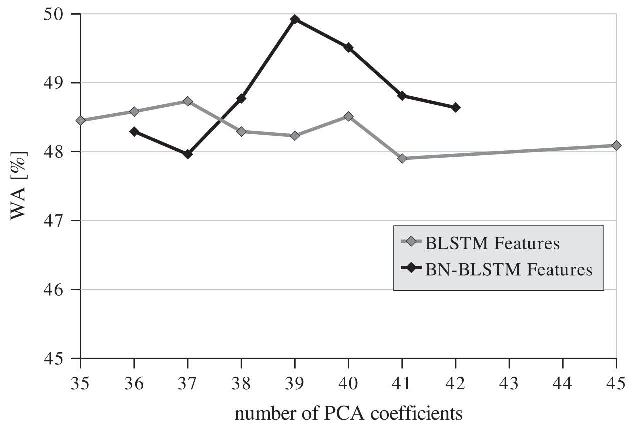

Fig. 8. Word accuracy (WA) on the COSINE test set as a function of the number of principal components; results are obtained using PCA-transformed feature vectors that contain logarithmized BLSTM activations and MFCC features.

Table 2
Framewise phoneme accuracies (FPA) and word accuracies (WA) for different recognition systems processing activations from the (third) hidden layer (Bottleneck), activations from the output layer (Tandem), discrete BLSTM phoneme predictions [6,14], or conventional MFCCs (HMM). Training and evaluation on the COSINE database or on the Buckeye corpus. Results for Bottleneck and Tandem systems are based on 39-dimensional feature vectors.

Model architecture	COSINE		Buckeye	
	FPA (%)	WA (%)	FPA (%)	WA (%)
Bottleneck BLSTM-HMM	69.96	**49.92**	69.89	**58.21**
Bottleneck LSTM-HMM	61.79	45.94	61.52	52.53
Bottleneck BRNN-HMM	56.93	41.39	53.40	49.28
Bottleneck RNN-HMM	48.88	40.74	47.05	48.78
Tandem BLSTM-HMM	69.96	48.23	69.89	57.80
Tandem LSTM-HMM	61.79	46.68	61.52	53.86
Tandem BRNN-HMM	56.93	40.67	53.40	48.64
Tandem RNN-HMM	48.88	40.14	47.05	48.21
Multi-stream BLSTM-HMM [6]	69.96	48.01	69.89	56.61
Discrete BLSTM feature [14]	66.41	45.04	69.89	55.91
HMM	56.91	43.36	53.20	50.97

The performance difference between the front-ends applying Bottleneck BLSTM features and BLSTM features derived from the output activations is statistically significant at the 0.002 level when using a z-test as described in [27]. For comparison, the last two rows of Table 2 again show the performance of the continuous–discrete BLSTM Tandem system introduced in [14] (45.04%) and the word accuracy of a baseline HMM processing only MFCC features (43.36%).

To collect further evidence for the obtainable ASR performance gains when applying the proposed Bottleneck-BLSTM front-end, we repeated our experiments, training and evaluating on the Buckeye corpus (see Section 4). Since the transcriptions of the Buckeye corpus also contain the events *laughter*, *noise*, *vocal noise*, and *garbage speech*, the size of the network output layers was increased by four from 41 to 45. Thus, we also increased the size of the third hidden layer from 80 to 90 to have roughly twice as many memory blocks as phoneme targets in the last hidden layer. Apart from that, no further modifications and optimizations (such as adaptation of the number of PCA coefficients) were carried out, aiming for a realistic performance assessment that shows how well a system designed for the COSINE task generalizes to other spontaneous speech scenarios. As shown in the last column of Table 2, the baseline HMM achieves a word accuracy of 50.97% which is comparable to the result reported in [25] (49.99%). Accuracies for the Buckeye experiment are notably higher than for the COSINE task since the Buckeye corpus contains speech which is less disfluent and noisy than in the COSINE database. Performance can be increased to up to 58.21% when applying our BN-BLSTM feature extraction. General trends are similar to the COSINE experiment: Again, the Bottleneck-BLSTM principle prevails over the BLSTM multi-stream approach employed in [6].

Finally, we examined whether part of the performance gap between RNN and BLSTM network architectures can be attributed to the higher number of trainable weights in the BLSTM networks rather than to the more effective context learning abilities of BLSTM front-ends. To this end, we trained an MLP that consists of three layers with sizes 321, 527 and 330, resulting in a network with 370 345 weights. The ratio of the sizes of the hidden layers is similar to the BLSTM network with the 78-128-80 hidden layer topology and the total number of weights is comparable to the BLSTM network applied for the COSINE experiment which has 369 249 weights. As the word accuracy obtained with the resulting MLP front-end is 42.72%, which is slightly lower than for the baseline HMM trained and evaluated on COSINE, we can conclude that simply increasing the size of the network does not lead to better recognition performance.

6. Conclusion

We showed how speech recognition in challenging scenarios can be improved by applying bidirectional Long Short-Term Memory modeling within the recognizer front-end. BLSTM networks are able to incorporate a flexible, self-learned amount of contextual information in the feature extraction process which was shown to result in enhanced probabilistic features, prevailing over conventional RNN or MLP features. In contrast to our earlier studies on BLSTM-based ASR systems, which exclusively used a discrete BLSTM phoneme estimate as additional feature, this paper investigated the benefit of generating feature vectors from the continuous logarithmized and PCA-transformed vector of BLSTM output activations. Fusing this concept with the Bottleneck technique enables the generation of a well decorrelated and compact feature space that carries information complementary to the original MFCC features. The experiments presented in this paper focused on the recognition of spontaneous, conversational, and partly disfluent, emotional, or noisy speech which usually leads to very poor ASR performance. Our BN-BLSTM technique is able to increase word accuracies from 43.36 to 49.92% and from 50.97 to 58.21% for the COSINE and the Buckeye task, respectively, and outperforms previous attempts to use BLSTM for continuous speech recognition as presented in a series of recent publications [14,15,6].

Future work should focus on hierarchical BLSTM topologies and on networks trained on phoneme state targets as alternative to phoneme targets. To exploit information from different probabilistic feature extractors, future studies should consider joint decoding of multiple phoneme prediction streams, e.g., via techniques for hybrid fusion such as asynchronous HMMs [28] or multi-dimensional Dynamic Time Warping [29]. Furthermore, BLSTM-based recognizer back-ends such as the Connectionist Temporal Classification technique [30] deserve attention in future ASR system development. Language modeling with BLSTM networks could be an effective way to enhance word-level context usage. Finally, we plan to develop an on-line version of the proposed (unidirectional) ASR front-end as an extension of the multi-stream ASR framework that is part of our real-time speech processing toolkit openSMILE [31].

Acknowledgments

The research leading to these results has received funding from the Federal Republic of Germany through the German Research Foundation (DFG) under Grant no. SCHU 2508/4-1.

References

[1] A. Hagen, A. Morris, Recent advances in the multi-stream HMM/ANN hybrid approach to noise robust ASR, Comput. Speech Lang. 19 (1) (2005) 3–30.
[2] B. Schuller, M. Wöllmer, T. Moosmayr, G. Rigoll, Recognition of noisy speech: a comparative survey of robust model architecture and feature enhancement, J. Audio Speech Music Process. ID 942617 (2009), http://dx.doi.org/10.1155/2009/942617.
[3] H. Hermansky, D.P.W. Ellis, S. Sharma, Tandem connectionist feature extraction for conventional HMM systems, in: Proceedings of ICASSP, Istanbul, Turkey, 2000, pp. 1635–1638.
[4] D.P.W. Ellis, R. Singh, S. Sivadas, Tandem acoustic modeling in large-vocabulary recognition, in: Proceedings of ICASSP, Salt Lake City, UT, USA, 2001, pp. 517–520.
[5] Q. Zhu, B. Chen, N. Morgan, A. Stolcke, Tandem connectionist feature extraction for conversational speech recognition, in: Machine Learning for Multimodal Interaction, Springer, 2005, pp. 223–231.
[6] M. Wöllmer, B. Schuller, G. Rigoll, Feature frame stacking in RNN-based Tandem ASR systems—learned vs. predefined context, in: Proceedings of Interspeech, Florence, Italy, 2011, pp. 1233–1236.
[7] F. Grezl, P. Fousek, Optimizing bottle-neck features for LVCSR, in: Proceedings of ICASSP, Las Vegas, NV, 2008, pp. 4729–4732.
[8] A. Graves, J. Schmidhuber, Framewise phoneme classification with bidirectional LSTM and other neural network architectures, Neural Netw. 18 (5-6) (2005) 602–610.
[9] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[10] F. Gers, J. Schmidhuber, F. Cummins, Learning to forget: continual prediction with LSTM, Neural Comput. 12 (10) (2000) 2451–2471.
[11] S. Fernandez, A. Graves, J. Schmidhuber, An application of recurrent neural networks to discriminative keyword spotting, in: Proceedings of ICANN, Porto, Portugal, 2007, pp. 220–229.
[12] M. Wöllmer, F. Eyben, A. Graves, B. Schuller, G. Rigoll, Bidirectional LSTM networks for context-sensitive keyword detection in a cognitive virtual agent framework, Cogn. Comput. 2 (3) (2010) 180–190.
[13] M. Wöllmer, B. Schuller, F. Eyben, G. Rigoll, Combining long short-term memory and dynamic bayesian networks for incremental emotion-sensitive artificial listening, IEEE J. Sel. Top. Signal Process. 4 (5) (2010) 867–881.
[14] M. Wöllmer, F. Eyben, B. Schuller, G. Rigoll, Recognition of spontaneous conversational speech using long short-term memory phoneme predictions, in: Proceedings of Interspeech, Makuhari, Japan, 2010, pp. 1946–1949.
[15] M. Wöllmer, F. Eyben, B. Schuller, G. Rigoll, A multi-stream ASR framework for BLSTM modeling of conversational speech, in: Proceedings of ICASSP, Prague, Czech Republic, 2011, pp. 4860–4863.

[16] F. Grezl, M. Karafiat, K. Stanislav, J. Cernocky, Probabilistic and bottle-neck features for LVCSR of meetings, in: Proceedings of ICASSP, Honolulu, Hawaii, 2007, pp. 757–760.
[17] A. Stupakov, E. Hanusa, D. Vijaywargi, D. Fox, J. Bilmes, The design and collection of COSINE, a multi-microphone in situ speech corpus recorded in noisy environments, Comput. Speech Lang. 26 (1) (2011) 52–66.
[18] M.A. Pitt, L. Dilley, K. Johnson, S. Kiesling, W. Raymond, E. Hume, E. Fosler-Lussier, Buckeye Corpus of Conversational Speech (2nd release), Department of Psychology, Ohio State University (Distributor), Columbus, OH, USA, 2007, ⟨www.buckeyecorpus.osu.edu⟩.
[19] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, in: S.C. Kremer, J.F. Kolen (Eds.), A Field Guide to Dynamical Recurrent Neural Networks, IEEE Press, 2001, pp. 1–15.
[20] M. Schuster, K.K. Paliwal, Bidirectional recurrent neural networks, IEEE Trans. Signal Process. 45 (1997) 2673–2681.
[21] M.A. Nicolaou, H. Gunes, M. Pantic, Continuous prediction of spontaneous affect from multiple cues and modalities in valence-arousal space, IEEE Trans. Affect. Comput. 2 (2011) 92–105.
[22] M. Liwicki, A. Graves, S. Fernandez, H. Bunke, J. Schmidhuber, A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks, in: Proceedings of ICDAR, Curitiba, Brazil, 2007, pp. 367–371.
[23] A. Graves, S. Fernandez, M. Liwicki, H. Bunke, J. Schmidhuber, Unconstrained online handwriting recognition with recurrent neural networks, Adv. Neural Inf. Process. Syst. 20 (2008) 1–8.
[24] M. Wöllmer, C. Blaschke, T. Schindl, B. Schuller, B. Färber, S. Mayer, B. Trefflich, On-line driver distraction detection using long short-term memory, IEEE Trans. Intell. Transp. Syst. 12 (2) (2011) 574–582.
[25] F. Weninger, B. Schuller, M. Wöllmer, G. Rigoll, Localization of non-linguistic events in spontaneous speech by non-negative matrix factorization and long short-term memory, in: Proceedings of ICASSP, Prague, Czech Republic, 2011, pp. 5840–5843.
[26] P. Werbos, Backpropagation through time: what it does and how to do it, Proceedings of the IEEE 78 (1990) 1550–1560.
[27] G.W. Snedecor, W.G. Cochran, Statistical Methods, 8th ed., Iowa State University Press, 1989.
[28] S. Bengio, An asynchronous Hidden Markov Model for audio-visual speech recognition, Advances in NIPS 15 (2003) 1–8.
[29] M. Wöllmer, M. Al-Hames, F. Eyben, B. Schuller, G. Rigoll, A multidimensional dynamic time warping algorithm for efficient multimodal fusion of asynchronous data streams, Neurocomputing 73 (1–3) (2009) 366–380.
[30] A. Graves, S. Fernandez, F. Gomez, J. Schmidhuber, Connectionist temporal classification: labelling unsegmented data with recurrent neural networks, in: Proceedings of ICML, Pittsburgh, USA, 2006, pp. 369–376.
[31] F. Eyben, M. Wöllmer, B. Schuller, openSMILE—the Munich versatile and fast open-source audio feature extractor, in: Proceedings of ACM Multimedia, Firenze, Italy, 2010, pp. 1459–1462.

Martin Wöllmer works as a researcher at the Technische Universität München (TUM). He obtained his diploma in Electrical Engineering and Information Technology from TUM where his current research activity includes the subject areas of pattern recognition and speech processing. His focus lies on automatic recognition of emotional and noisy speech, multimodal data fusion, affective computing, and context-sensitive machine learning. His reviewing engagement includes several journals such as Speech Communication and Computer Speech and Language. Publications of his in various journals and conference proceedings cover novel and robust modeling architectures such as Switching Linear Dynamic Models and Long Short-Term Memory networks.

Björn Schuller received his diploma and doctoral degrees in Electrical Engineering and Information Technology from TUM, where he currently stays as lecturer in Pattern Recognition. He authored more than 250 publications in books, journals, and conference proceedings. Best known are his works advancing Audiovisual Processing in the areas of Affective Computing and Multimedia Retrieval. He serves as member of the steering committee of the IEEE Transactions on Affective Computing, as an associate editor and reviewer for several journals, including Elsevier Signal Processing and Speech Communication, and as invited speaker, session organizer and chairman, and programme committee member of numerous international conferences.