Feature Frame Stacking in RNN-based Tandem ASR Systems – Learned vs. Predefined Context

Martin Wöllmer, Björn Schuller, Gerhard Rigoll

Institute for Human-Machine Communication, Technische Universität München, Germany

[woellmer,schuller,rigoll]@tum.de

Abstract

As phoneme recognition is known to profit from techniques that consider contextual information, neural networks applied in Tandem automatic speech recognition (ASR) systems usually employ some form of context modeling. While approaches based on multi-layer perceptrons or recurrent neural networks (RNN) are able to model a predefined amount of context by simultaneously processing a stacked sequence of successive feature vectors, bidirectional Long Short-Term Memory (BLSTM) networks were shown to be well-suited for incorporating a self-learned amount of context for phoneme prediction. In this paper, we evaluate combinations of BLSTM modeling and frame stacking to determine the most efficient method for exploiting context in RNN-based Tandem systems. Applying the COSINE corpus and our recently introduced multi-stream BLSTM-HMM decoder, we provide empirical evidence for the intuition that BLSTM networks redundantize frame stacking while RNNs profit from predefined feature-level context.

Index Terms: context modeling, long short-term memory, recurrent neural networks, automatic speech recognition

1. Introduction

In contrast to the automatic recognition of well-articulated read speech or defined sets of command words which works well with standard techniques such as Hidden Markov Models (HMM), the recognition of disfluent, spontaneous, emotionally colored, and noisy conversational speech demands for novel techniques that go beyond state-of-the-art approaches. For example, so-called Tandem automatic speech recognition (ASR) systems, that apply multi-layer perceptrons (MLP) or recurrent neural networks (RNN) which generate phoneme prediction features for HMM-based decoding have shown enhanced performance in challenging conditions [1, 2, 3]. Both, neural networks employed in Tandem systems and HMM decoders rely on effective context modeling in order to capture co-articulation effects in human speech as well as transition probabilities between states, phonemes, and words. Conventional ASR systems model context on multiple levels, including feature-level context by appending delta features, mid-level context via Markov assumptions during phoneme modeling, and high-level context by the use of language models. Additionally, MLPs or RNNs for phoneme estimation usually consider context by simultaneously processing successive stacked feature vectors [2, 4].

An efficient method to model a flexible amount of contextual information within recurrent neural networks is the so-called Long Short-Term Memory (LSTM) architecture [5]. LSTM networks offer the advantage that the amount of context that is relevant for the classification task is learned during training and does not have to be manually specified. Bidirectional Long Short-Term Memory (BLSTM) networks consider both, past and future context and were successfully applied for phoneme recognition [6], outperforming conventional RNN approaches as well as triphone HMMs. In successive studies on BLSTM speech processing, excellent results for keyword spotting tasks were reported [7, 3]. Recently, BLSTM networks incorporated into Tandem continuous speech recognition systems showed good performance [8].

In this paper, we investigate how feature frame stacking affects the performance of LSTM-based phoneme recognition and Tandem ASR, aiming to determine whether learned or predefined context leads to better accuracies. We evaluate different bi- and unidirectional network architectures with and without Long Short-Term Memory employing varying ranges of predefined feature-level context. BLSTM networks are evaluated as part of our recently proposed multi-stream BLSTM-HMM system for continuous speech recognition in challenging conditions [8]. Further, we show architectural enhancements of our multi-stream approach, leading to improved word accuracies when tested on the COSINE database [9] which contains spontaneous, conversational, and partly noisy speech.

We explain the concept of LSTM, bidirectional networks, and multi-stream BLSTM-HMM decoding in Section 2. Feature frame stacking is briefly outlined in Section 3 before experimental results are detailed in Section 4.

2. BLSTM Context Modeling

2.1. Long Short-Term Memory

A simple and widely used technique for context-sensitive sequence labeling based on neural networks is the application of *recurrent* neural networks. RNNs are able to model a certain amount of context by using cyclic connections and can in principle map from the entire *history* of previous inputs to each output. Yet, the analysis of the error flow in conventional recurrent neural nets resulted in the finding that long-range context is inaccessible to standard RNNs since the backpropagated error either blows up or decays over time (vanishing gradient problem [10]). Thus, only context sizes in the order of 10 frames can be captured via conventional RNNs [6]. One of the most effective techniques to overcome the vanishing gradient problem is the Long Short-Term Memory architecture [5], which is able to store information in linear memory cells over a longer period of time and can learn the optimal amount of contextual information relevant for the classification task. 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.

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

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

and

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

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 (see equations 3 and 7). 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, while I, H, and C are the number of inputs, the number of cells in the hidden layer, and the number of memory cells in one block. 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:

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

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

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

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

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

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

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:

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

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

Finally, the memory cell output is determined as

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

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.

In recent years, the LSTM technique has been successfully applied for a variety of pattern recognition tasks, including phoneme classification [6], keyword spotting [3], emotion recognition [11], and handwriting recognition [12].

2.2. Bidirectional Networks

Another shortcoming of standard RNNs is that they have access to past but not to future context. This can be overcome by using *bidirectional* RNNs [13], 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. 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, forward and backward context can be used during decoding.

2.3. Multi-Stream BLSTM-HMM Decoding

To exploit LSTM-based phoneme recognition for continuous speech recognition, we apply our recently introduced multi-stream BLSTM-HMM technique [8], which decodes both, low-level features, and BLSTM phoneme estimates. In every time frame t the HMM uses two independent observations: the MFCC features x_t and the BLSTM phoneme prediction feature b_t. The vector x_t also serves as input for the BLSTM, whereas the size of the BLSTM input layer corresponds to the dimensionality of the acoustic feature vector. The vector of BLSTM output activations o_t contains one probability score for each of the P different phonemes at each time step. b_t is the index of the most likely phoneme:

$$b_t = \arg\max_j (o_{t,1}, ..., o_{t,j}, ..., o_{t,P}). \quad (10)$$

In every time step the BLSTM generates a phoneme prediction according to Equation 10 and the HMM models $x_{1:T}$ and $b_{1:T}$ as two independent data streams. With $y_t = [x_t; b_t]$ being the joint feature vector consisting of continuous MFCC and discrete BLSTM observations and the variable a denoting the stream weight of the first stream (i. e., the MFCC stream), the multi-stream HMM emission probability while being in a certain state s_t can be written as

$$p(y_t|s_t) = \left[\sum_{m=1}^{M} c_{s_t m} \mathcal{N}(x_t; \mu_{s_t m}, \Sigma_{s_t m}) \right]^a \times p(b_t|s_t)^{2-a}. \quad (11)$$

Thus, the continuous MFCC observations are modeled via a mixture of M Gaussians per state while the BLSTM prediction is modeled using a discrete probability distribution $p(b_t|s_t)$. The index m denotes the mixture component, $c_{s_t m}$ is the weight of the m'th Gaussian associated with state s_t, and $\mathcal{N}(\cdot; \mu, \Sigma)$ represents a multivariate Gaussian distribution with mean vector μ and covariance matrix Σ. The distribution $p(b_t|s_t)$ is trained to model typical phoneme confusions that occur in the BLSTM network.

A real-time implementation of our multi-stream decoder is publicly available as part of our on-line speech processing toolbox openSMILE [14].

3. Feature Frame Stacking

A straightforward method to model temporal context within neural networks is to stack a fixed number of n successive frames, so that a *sequence* of feature vectors is presented to the

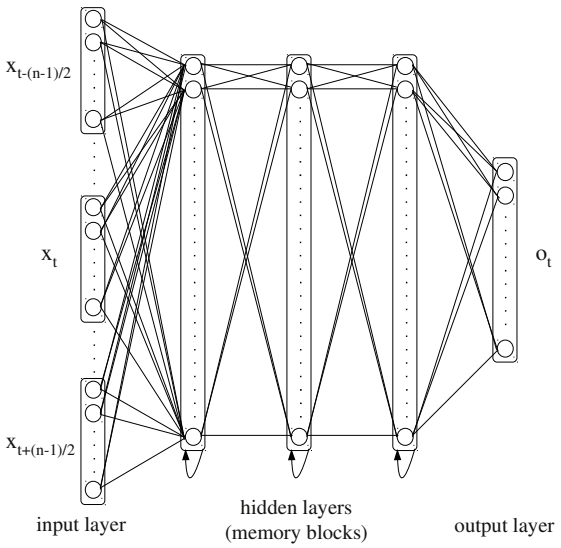

Figure 1: *Example of a neural network processing n stacked feature frames.*

network at each time step. In MLP-based Tandem ASR systems, it is common to stack an equal number of past and future feature frames around the central feature vector x_t. Thus, a sliding window from $t-(n-1)/2$ to $t+(n-1)/2$ is applied to merge n successive feature vectors of size N to an $n \cdot N$ dimensional extended feature vector x'_t, i. e.,

$$x'_t = [x_{t-\frac{n-1}{2}}; \dots x_t; \dots x_{t+\frac{n-1}{2}}] \quad \text{for} \quad \frac{n-1}{2} < t \leq T - \frac{n-1}{2}. \tag{12}$$

In order to obtain valid vectors for $t \leq (n-1)/2$ and $t > T-(n-1)/2$, the first and the last feature vector of $x_{1:T}$ has to be copied $(n-1)/2$ times. Figure 1 shows a schematic example of a network processing n frames to produce a vector of output activations o_t at time t. The network consists of three hidden layers, an input layer of size $n \cdot N$ and an output layer of size P.

4. Experiments and Results

4.1. The COSINE Corpus

All experiments presented in this paper are speaker-independent and were carried out using the 'COnversational Speech In Noisy Environments' (COSINE) corpus [9] which is a relatively new database containing multi-party conversations recorded in real world environments. The 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 using multiple microphones simultaneously, however, to match most application scenarios, we exclusively used speech recorded by a close-talking microphone (Sennheiser ME-3).

We used all ten transcribed sessions, containing 11.40 hours of pairwise conversations and group discussions. All 37 speakers are fluent, but not necessarily native English speakers. Each speaker participated in only one session and the speakers' ages range from 18 to 71 years (median 21 years).

For our experiments, we used the recommended test set (sessions 3 and 10) which comprises 1.81 hours 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 is 4.8 k, whereas the out-of-vocabulary (OOV) rate in the test set is 3.4 %.

4.2. Network Training

All networks were trained on framewise phoneme targets obtained via HMM-based forced alignment of the COSINE training set. As network input x_t we used MFCCs 1 to 12 including log. energy together with first and second order regression coefficients, i. e., feature vectors of size 39. For feature frame stacking, we evaluated sliding windows of lengths up to $n = 9$, which is typical for Tandem ASR systems [2]. This corresponds to stacked feature vectors of size 351. To compensate for stationary noise effects, we applied cepstral mean normalization. We evaluated four different network architectures: conventional recurrent neural networks, bidirectional neural networks (BRNN), unidirectional LSTM networks, and bidirectional LSTM networks. All networks consisted of three hidden layers (per input direction) with a size of 78, 128, and 80 hidden units, respectively. Each LSTM memory block contained one memory cell. The networks were trained on the standard (CMU) set of 39 different English phonemes with additional targets for *silence* and *short pause*. Training was aborted as soon as no improvement on the validation set (sessions 1 and 8) could be observed for at least 50 epochs, and we chose the network that achieved the best framewise phoneme error rate on the validation set.

4.3. Framewise Phoneme Recognition

Table 1 shows the framewise phoneme error rates when applying different neural network architectures and stack sizes of 1 to 9 feature frames. For bidirectional LSTM networks the error rate increases from 30.04 % to 32.02 % as more successive frames are simultaneously processed. Hence, BLSTM networks seem to learn context better if feature frames are presented one by one and the increased size of the input layer rather harms recognition performance. For unidirectional LSTM networks we observe a different trend: The error rate slightly decreases from 38.21 % to 37.43 % as more frames are processed. This is most likely due to the (small amount of) *future* context information which is available to the LSTM networks if stacking is used and which is not available for fully causal LSTMs observing only one frame per time step. Still, the error rate is notably lower for BLSTM networks. In contrast to BLSTM networks, both, BRNNs and RNNs profit from feature frame stacking: error rates decrease from 43.07 % to 41.83 % and from 51.12 % to 48.82 %, respectively. This indicates that even though recurrent networks can model a limited amount of context, it is beneficial to introduce a predefined amount of temporal context in the form of stacked feature vectors. However, if we compare the performance of LSTM and RNN architectures, we see that learned LSTM long-range context prevails over feature frame stacking.

Table 1: *Framewise phoneme error rates (FER) on the COSINE validation and test set using different network architectures and stack sizes of 1 to 9 frames.*

FER [%]	**number of frames**				
network type	1	3	5	7	9
validation set					
BLSTM	32.27	32.81	33.42	33.79	34.29
LSTM	40.57	40.84	40.76	40.50	40.26
BRNN	43.62	42.84	43.65	44.05	43.17
RNN	52.18	52.29	50.94	51.40	51.22
test set					
BLSTM	**30.04**	30.86	31.89	31.84	32.02
LSTM	38.21	38.36	37.81	37.67	**37.43**
BRNN	43.07	41.97	42.39	43.04	**41.83**
RNN	51.12	50.47	49.21	49.47	**48.82**

Table 2: *Word accuracies on the COSINE test set when applying different multi- and single-stream systems with three hidden LSTM layers (L-L-L) or with one backpropagation and two LSTM layers (B-L-L) and different frame stack sizes (# fr.).*

system architecture	**hidden layers**	**# fr.**	**WA [%]**
multi-str. BLSTM-HMM	L-L-L	1	**48.01**
multi-str. BLSTM-HMM	L-L-L	9	47.17
multi-str. BLSTM-HMM [8]	B-L-L	1	46.50
single-str. BLSTM-HMM [15]	B-L-L	1	45.04
triphone HMM	-	1	43.36

4.4. Tandem Speech Recognition

Applying the multi-stream BLSTM-HMM system outlined in Section 2.3, we evaluated the word accuracy on the COSINE test set when using the network type with the best framewise phoneme error rate (i. e., the BLSTM architecture). The underlying HMM system was configured as in [8] and the stream weight variable was set to $a = 1.1$. Starting from the multi-stream system presented in [8], which used a standard backpropagation layer as first hidden layer in the BLSTM network, we found that replacing the backpropagation layer with a third LSTM layer increases the word accuracy (WA) from 46.50 % to 48.01 %. The multi-stream system prevails over the single-stream Tandem approach introduced in [15] (WA of 45.04 %) and outperforms standard triphone HMMs using only MFCC vectors as observations (WA of 43.36 %). As observed for framewise phoneme classification (Section 4.3), feature frame stacking leads to less accurate phoneme estimates if BLSTM networks are applied. This results in a decrease of the word accuracy for continuous speech recognition (WA of 47.17 % for stack size $n = 9$).

5. Conclusion

We investigated different methods to model contextual information in neural networks for enhanced RNN-based phoneme classification and Tandem speech recognition, considering both, predefined and learned context. Exploiting a self-learned amount of contextual information, the concept of Long Short-Term Memory RNNs allows for co-articulation and temporal context modeling via memory blocks in the hidden layer and was shown to enable the highest phoneme recognition accuracies. Combining bidirectional LSTM and feature frame stacking as a simple method to incorporate a predefined amount of context resulted in increased error rates while unidirectional LSTM profit from frame stacking due to the incorporation of future feature frames. Conventional RNN architectures reach lower error rates when stacking is used but cannot compete with the phoneme recognition accuracies of LSTM-based recognizers.

A promising direction for future research is the combination of the bottleneck feature compression technique [2] and LSTM networks.

6. Acknowledgements

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.

7. References

[1] Q. Zhu, B. Chen, N. Morgan, and A. Stolcke, "Tandem connectionist feature extraction for conversational speech recognition," in *Machine Learning for Multimodal Interaction*. Springer, 2005, pp. 223–231.

[2] F. Grezl and P. Fousek, "Optimizing bottle-neck features for LVCSR," in *Proc. of ICASSP*, Las Vegas, NV, 2008, pp. 4729–4732.

[3] M. Wöllmer, F. Eyben, A. Graves, B. Schuller, and G. Rigoll, "Bidirectional LSTM networks for context-sensitive keyword detection in a cognitive virtual agent framework," *Cognitive Computation*, vol. 2, no. 3, pp. 180–190, 2010.

[4] S. Thomas, S. Ganapathy, and H. Hermansky, "Phoneme recognition using spectral envelope and modulation frequency features," in *Proc. of ICASSP*, Taipei,Taiwan, 2009, pp. 4453–4456.

[5] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[6] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5-6, pp. 602–610, 2005.

[7] S. Fernandez, A. Graves, and J. Schmidhuber, "An application of recurrent neural networks to discriminative keyword spotting," in *Proc. of ICANN*, Porto, Portugal, 2007, pp. 220–229.

[8] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "A multi-stream ASR framework for BLSTM modeling of conversational speech," in *Proc. of ICASSP*, Prague, Czech Republic, 2011.

[9] A. Stupakov, E. Hanusa, D. Vijaywargi, D. Fox, and J. Bilmes, "The design and collection of COSINE, a multi-microphone in situ speech corpus recorded in noisy environments," *Computer Speech and Language*, 2011, to appear.

[10] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: the difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, S. C. Kremer and J. F. Kolen, Eds. IEEE Press, 2001, pp. 1–15.

[11] M. Wöllmer, B. Schuller, F. Eyben, and G. Rigoll, "Combining long short-term memory and dynamic bayesian networks for incremental emotion-sensitive artificial listening," *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 5, pp. 867–881, 2010.

[12] A. Graves, S. Fernandez, M. Liwicki, H. Bunke, and J. Schmidhuber, "Unconstrained online handwriting recognition with recurrent neural networks," *Advances in Neural Information Processing Systems*, vol. 20, pp. 1–8, 2008.

[13] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, pp. 2673–2681, 1997.

[14] F. Eyben, M. Wöllmer, and B. Schuller, "openSMILE - the Munich versatile and fast open-source audio feature extractor," in *Proc. of ACM Multimedia*, Firenze, Italy, 2010, pp. 1459–1462.

[15] M. Wöllmer, F. Eyben, B. Schuller, and G. Rigoll, "Recognition of spontaneous conversational speech using long short-term memory phoneme predictions," in *Proc. of Interspeech*, Makuhari, Japan, 2010, pp. 1946–1949.