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.


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. Bidirec-
operations within a memory block, an LSTM block can be interpreted as (differentiable) memory chip in a digital computer. If $\alpha^t_i$ denotes the activation of the input gate at time $t$ before the activation function $f_\alpha$ has been applied and $\beta^t_i$ represents the activation after application of the activation function, the input gate activations (forward pass) can be written as

$$\alpha^t_i = \sum_{i=1}^{I} \eta^i x_i^t + \sum_{h=1}^{H} \eta^h j_i^h + \sum_{c=1}^{C} \eta^c o_i^c s_{t-1}$$

and

$$\beta^t_i = f_\beta(\alpha^t_i),$$

respectively. The variable $\eta^i$ 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 $x_i^t$, 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 $f_\alpha$ can be calculated as follows:

$$\alpha^t_f = \sum_{i=1}^{I} \eta^i f_i x_i^t + \sum_{h=1}^{H} \eta^h f_h \beta_{t-1}^h + \sum_{c=1}^{C} \eta^c f_c s_{t-1}^c$$

and

$$\beta^t_f = f_\beta(\alpha^t_f).$$

The memory cell output is determined as

$$s_t^c = \beta_t^c f_t s_{t-1}^c + \beta_t^c f_t (\alpha_t^c).$$

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:

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

In every time step the BLSTM generates a phoneme prediction according to Equation 10 and the HMM models $x_t; \tau$ and $b_t; \tau$ 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

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

\[
x'_t = [x_{t-n+1}; \ldots; x_t; \ldots; x_{t+n-1}],
\]

for \( n-1 \leq t \leq T - n-1 \).

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.

<table>
<thead>
<tr>
<th>network type</th>
<th>validation set</th>
<th>test set</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>number of frames</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>BLSTM</td>
<td>32.27</td>
<td>32.81</td>
</tr>
<tr>
<td>LSTM</td>
<td>40.57</td>
<td>40.84</td>
</tr>
<tr>
<td>BRNN</td>
<td>43.62</td>
<td>42.84</td>
</tr>
<tr>
<td>RNN</td>
<td>52.18</td>
<td>52.29</td>
</tr>
</tbody>
</table>

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.).

<table>
<thead>
<tr>
<th>system architecture</th>
<th>hidden layers</th>
<th># fr.</th>
<th>WA [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>multi-str. BLSTM-HMM</td>
<td>L-L-L</td>
<td>1</td>
<td>48.01</td>
</tr>
<tr>
<td>multi-str. BLSTM-HMM</td>
<td>L-L-L</td>
<td>9</td>
<td>47.17</td>
</tr>
<tr>
<td>multi-str. BLSTM-HMM [8]</td>
<td>B-L-L</td>
<td>1</td>
<td>46.50</td>
</tr>
<tr>
<td>triphone HMM</td>
<td>-</td>
<td>1</td>
<td>43.36</td>
</tr>
</tbody>
</table>

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