ACOUSTIC MODELLING WITH CD-CTC-SMBR LSTM RNNS

Andrew Senior, Haşim Sak, Félix de Chaumont Quitry, Tara Sainath, Kanishka Rao

Google

{hasim,andrewsenior,fcq,tsainath,kanishkarao}@google.com

ABSTRACT

This paper describes a series of experiments to extend the application of Context-Dependent (CD) long short-term memory (LSTM) recurrent neural networks (RNNs) trained with Connectionist Temporal Classification (CTC) and sMBR loss. Our experiments, on a noisy, reverberant voice search task, include training with alternative pronunciations and the application to child speech recognition; combination of multiple models, and convolutional input layers. We also investigate the latency of CTC models and show that constraining forward-backward alignment in training can reduce the delay for a real-time streaming speech recognition system. Finally we investigate transferring knowledge from one network to another through alignments.

Index Terms: Long Short Term Memory, Recurrent Neural Networks, Connectionist Temporal Classification, sequence discriminative training, knowledge transfer.

1. INTRODUCTION

In the last few years, most state-of-the-art automatic speech recognition systems have used neural network acoustic models to estimate probabilities which are aggregated in a hidden Markov model "decoder". Recently, recurrent neural networks (RNNs), and in particular deep Long Short Term Memory (LSTM) RNNs [1] have been shown to outperform deep neural networks (DNNs) [2]. Most recently [3] we have shown that greater accuracy can be obtained with models with a "blank" symbol that are trained using the connectionist temporal classification (CTC [4]) algorithm followed by sequence discriminative training, and using context dependent whole-phone models [5].

LSTM RNNs are a variant of recurrent neural networks proposed [6] as a way to circumvent the vanishing gradient problem and enable the propagation of gradients over long time spans and hence learn longer-term dependencies than are feasible with conventional RNNs. An LSTM layer consists of multiple memory cells which can store a scalar state over time and with three gates that control whether new information is added to the cell, whether the cell state should be forgotten and whether the cell state should be passed forward to the next layer in a network. LSTM layers can be stacked to give deep architectures which allow multiple nonlinear operations for a single time-step, though because of their recurrent nature, their effective depth increases with greater time offsets between inputs and outputs.

Graves *et al.* proposed CTC as a way to train recurrent networks on sequences of symbols where no alignment is given. An additional "blank" output is permitted to enable sequences of input data longer than the corresponding label sequence, as is often the case in speech or handwriting recognition. In contrast to conventional alignment whereby every frame is given a label from the target sequence, and the labels indicate the segmentation of the sequence with repeated labels indicating longer durations, with CTC an output may only be high for a single frame to indicate the presence of the symbol, with other frames labelled "blank," and duration information is discarded. During training CTC constantly aligns every sequence and trains to maximize the total probability of all valid label sequences. Because of the memory of the LSTM model this means that the outputs no longer need to occur at the same time as the input features to which they correspond.

In our previous work [2] we have shown that models with a blank symbol that are initialized with CTC can be improved upon with sMBR sequence-discriminative training. We then showed [3] that such models, using long-duration features (95ms of speech represented as 8 stacked overlapping log-mel filterbank features, generated with a 25ms window FFT every 10ms), downsampled and processed every 30ms, can outperform conventionally-trained LSTM models when using context dependent phone targets [5]. We use the term CD-CTC-sMBR LSTM RNN for these models.

In this paper we present a number of extensions and refinements to our CD phone CTC models. Section 3 describes our task, data and the baseline model we described previously. Thereafter each section presents one idea with related research and our experiments and results. Section 2.3 describes how alternative pronunciations can be successfully handled within CTC training. Section 2.2 describes improved performance on noise-corrupted and child speech data. Sections 3.1 and 3.2 demonstrate improved inference and decoding speed with our low-frame-rate CTC models, and show how constraints during training can limit latency in decoding. Section 3.3 describes the use of convolutional input layers for CTC stacked frames. Section 3.4 describes experiments in CTC model combination and section 3.5 shows knowledge transfer between CTC models using alignments.

2. EXPERIMENTAL SET-UP

We evaluate speech recognition performance with acoustic models trained with 9287 context dependent phone models with a blank symbol. The models are initially trained from scratch using the CTC algorithm to constantly realign with the Baum-Welch algorithm and trained using a cross-entropy loss. Models are then further trained sequence-discriminatively using the sMBR loss.

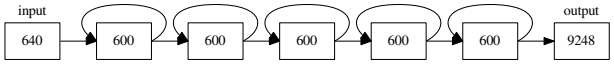


Fig. 1: Layer connections in unidirectional 5-layer LSTM RNNs.

The principal model that we investigate, shown in Figure 1, is the one from our most recent work [3] which has 5 LSTM lay-

Model	Description	Adult WER (%)		Child WER (%)	
		Clean	Noisy	Clean	Noisy
CTC	Train on adult data	11.5	13.0	12.0	14.0
CTC	Train on child + adult data (with flatstart)	11.2	12.6	9.9	11.3
CLDNN	Train on child + adult data	11.9	13.5	9.9	12.6

Table 1: *WERs for three different sMBR-trained models on clean & noise-corrupted versions of adult and child test sets.*

ers of 600 cells, each with its own gates. The output distribution was the same 9288 context dependent phone set + blank used previously, and the inputs are gain 80-dimensional log-mel filterbank energies computed on 25ms window every 10ms, stacked 8-deep and downsampled by a factor of 3 (i.e. one stacked-frame every 30ms, with 65ms of overlap). In this paper we do not investigate context-independent phone models, since they did not perform as well as context-dependent phone models. Nor do we investigate bidirectional models which can not be used in a streaming speech recognition system.

Single-pass decoding is carried out with a conventional WFST-based decoder using a 100-million 5-gram language model and a vocabulary larger than 5 million words.

2.1. Google Now task

We carried out experiments on data from the Google Now voice search speech recognition task in US English. The approximately 2000 hours of training data consists of 3 million anonymized utterances of live 16kHz traffic. These are corrupted using a room simulator which adds artificial noise (non-speech audio from YouTube videos) and reverberation. We generate 20 different corrupted versions of each utterance. The test set consists of 28,000 utterances of similar traffic, either clean or corrupted with a similar distribution of reverberation and noise levels, but with a held-out noise data.

2.2. Child speech task

We have recently described [7] the development of a speech recognition system for "YouTube Kids" an application specifically for children with a speech recognition interface which enables search for children who are too young to read or type. As part of this effort, we collected a database of 1.9M anonymized "high-pitched" US English utterances most of which are believed to be from children. We added this data to our "adult" speech database of 3M utterances, again perturbing each with 20 different noise / reverberation combinations. A similar held out set is used as a test set, with and without noise-corruption.

2.3. Flat start

In English, there are many homographs — words with alternative pronunciations for a given written form. When starting from a written transcription and training a spoken-form model, it is necessary to choose which spoken form to use. In conventional training we can apply Viterbi alignment to a lattice containing alternative pronunciations and allow the model to choose. Hitherto we have trained CTC models using a unique alignment string which in practice was derived from an alignment with an earlier (DNN) model. For the experiments in this paper, we have found that we can apply the forward-backward algorithm to the full CD phone lattice and can jointly train a CTC model and choose the alternative pronunciation.

3. EXPERIMENTS

Table 1 shows word error rates of sMBR-trained models on clean and noise-corrupted versions of adult and child test sets. The CTC LSTM network trained on adult and child speech performs better on adult test data and significantly better on child data than one trained only on adult data. In most cases, the CTC LSTM performs better than a CLDNN [8] trained on the same data.

3.1. Speed

Figure 2 shows a comparison of word error rate (WER) against 90th percentile real-time factor (time to process an utterance divided by the duration of the utterance) for a CTC LSTM and a CLDNN model obtained by changing the beam width while keeping maximum number of arcs at 8000 in decoding. The CTC model uses the context dependent phones and operates at 33 frames/s rate. The CLDNN model uses the HMM states (3 states for each context dependent phone) and operates at 100 frames/sec rate. While the accuracy difference between these models is relatively small, the decoding speed is about 3 times faster for the CTC model than the conventional CLDNN model due to significantly reduced frame rates. We also observed that due to spiky predictions from the CTC model, we can constrain the search space more than conventional models without hurting the recognition accuracy by limiting the maximum number of arcs in decoding.

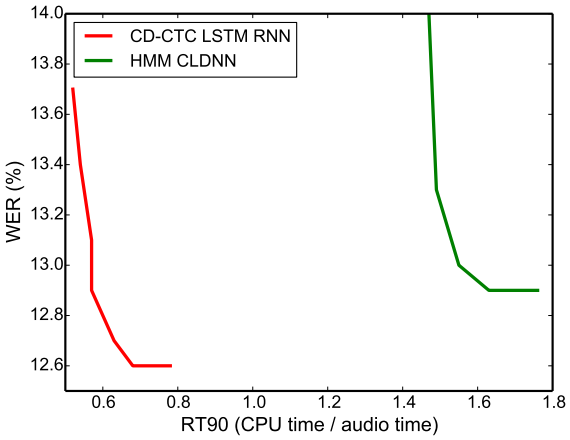


Fig. 2: 90th percentile real-time factor vs WER for CTC and conventional models.

3.2. Delay constraints

In training conventional recurrent neural network models, it is common to derive the labels from a forced alignment, but to choose a

time delay [9] between the acoustic frame presentation and the label output to give the network future acoustic context on which to base its predictions, akin to the use of a future context window in the frame stacking for GMM or DNN models. Such a delay is typically around 5 frames or 50ms. With CTC, there is no time alignment supervision since the network is constantly integrating over all possible alignments. This means that the LSTM can vary the delay between acoustics and outputs, using an arbitrarily large future context if that helps optimizing the total sequence probability. In practice, as shown in Figure 3 (top), the network does delay the outputs considerably with respect to the alignment of a DNN.

This delay induces latency in the decoding of speech. Google Now's speech recognition is a live streaming service where intermediate results are displayed while the user is still speaking. Additional latency from CTC self-alignment is undesirable, so we investigated applying constraints on the CTC alignment to reduce the delay. Delay can be limited by restricting the set of search paths used in the forward-backward algorithm to those in which the delay between CTC labels and the "ground truth" alignment does not exceed some threshold. Figure 3 shows a set of alignments with models trained with different delay constraints. Table 2 shows that tightening the constraint in CTC training degrades the WER, but after sequence training the performance with and without the constraint is similar.

Training	CTC WER (%)	sMBR WER (%)
no constraint	14.3	13.0
300ms delay	14.5	
200ms delay	14.7	
150ms delay	14.6	
100ms delay	14.8	13.0
60ms delay	15.0	

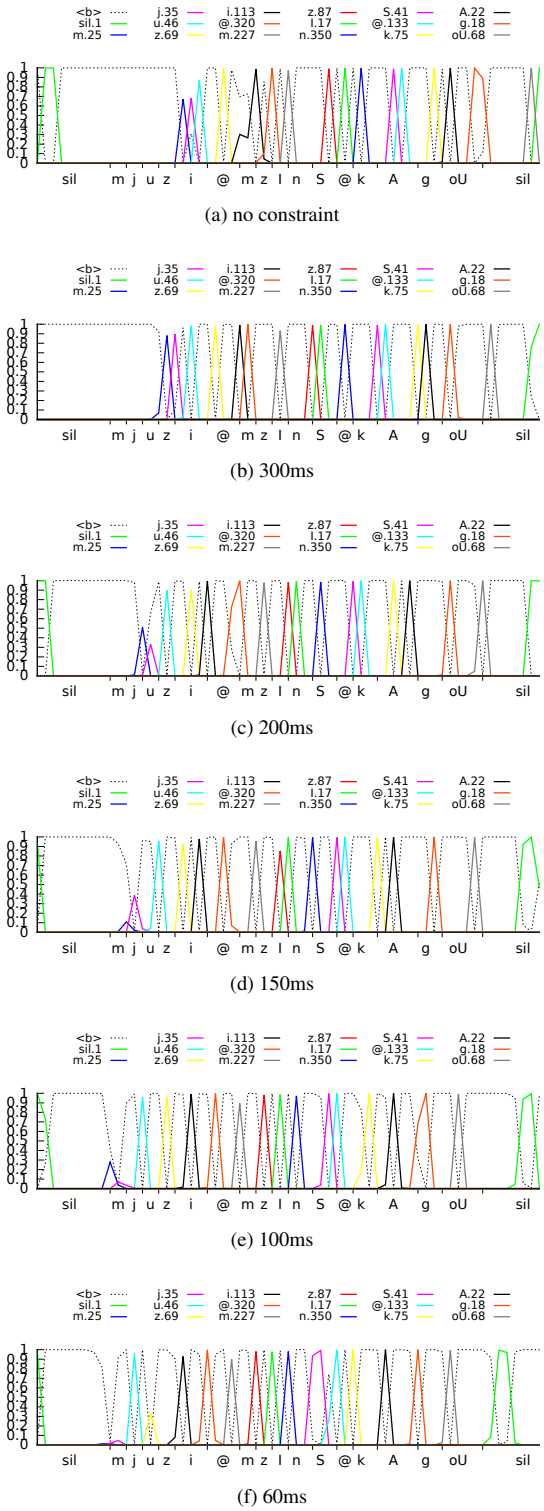
Table 2: *WERs for models trained with different delay constraints, with and without sMBR training.*

3.3. Convolution

Sainath *et al.* [8] recently described a deep network architecture which combines convolutional layers followed by LSTM layers followed by fully connected layers and finishing with a softmax layer. They termed this the "CLDNN" (for convolution + LSTM + DNN) and showed improved results compared to deep LSTM architectures. This naturally leads us to conjecture that a similar architecture trained with CTC would lead to improved performance compared to a deep LSTM CTC network.

For all our experiments here, we retain the 5 LSTM layer + softmax architecture, and simply precede the LSTM layer with a rectified linear convolution layer followed by max pooling and linear dimensionality reduction layer. When operating on s stacked frames of 80-dimensional filterbanks, the N filters we use have a support of $15 \times s$ support, convolved in frequency with a step of 1, with non-overlapping max-pooling across 6 frequency bands. This results in $22 \times N$ activations which are linearly projected to 256 dimensions for input to the first LSTM layer. This process is shown in Figure 4. For our experiments here, we have used $N = 96$

A variety of other approaches are feasible, in particular performing 15×1 convolutions with shared parameters separately on each of the stacked frames, but so far results with this approach have not performed as well. Table 3 shows that convolutions with 3, 5 or 8 stacked features perform similarly, and perform slightly worse than fully-connected inputs.



(a) no constraint

(b) 300ms

(c) 200ms

(d) 150ms

(e) 100ms

(f) 60ms

Fig. 3: Label posteriors estimated by CD-CTC LSTM RNN models trained with different delay constraints plotted against fixed DNN frame level alignments shown only for labels in the alignment on a held out utterance *'museums in Chicago'*. <b> refers to the *blank* label.

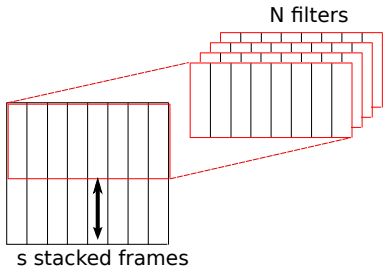


Fig. 4: Convolution of N filters in frequency across s frames, where the filters are s-frames wide.

Model	Description	WER (%)	
		CTC	sMBR
A	fully-connected 8 frames	14.4	12.6
B	fully-connected 8 frames	14.2	12.6
C	Convolution on 3 frames	14.6	
D	Convolution on 5 frames	14.6	12.9
E	Convolution on 8 frames	14.8	
A & D	Combined CTC models	13.9	12.6
A & D	Combined sMBR models		12.5
A & B	Combined CTC models		12.7
A, B & D	Combined CTC models	14.4	12.8
A, B & D	ROVER		12.2

Table 3: *WERs for individual models and combinations after CTC or sMBR training (trained on adult + child data).*

3.4. Model combination

It is well-known that multiple classifiers can often be combined together to create a joint classifier which performs better than any of the original classifiers. The simplest method is to form a weighted combination of the classifiers' posteriors in *score fusion*. Such score combination techniques have been used for speech recognition, for instance we have seen around a 7% relative reduction in WER when combining 3 conventional LSTM classifiers trained under the same conditions but for randomization (of both weight initialization and data-shuffling).

The ROVER [10] technique has long been used to combine the output hypotheses of speech recognition systems, particularly when the systems have been developed independently, so share no intermediate representation (such as the CD state inventory) where score fusion could be carried out. At its simplest, ROVER implements a voting strategy across systems to combine alternative hypotheses for time segments. Alternatives have been proposed to use score and confidence measures for N-best lists or lattices.

Since CTC networks with 30ms features use so little computation for acoustic model computation and search (Section 3.1), model combination is an attractive option. If we can get further gains by combining three models, we can achieve this while still being no slower than a conventional LSTM acoustic model.

ROVER is directly applicable to CTC networks and can even be used to combine CTC and conventional systems (e.g. DNN, LSTM, CLDNN) — we decode separately with each of our candidate networks, and use ROVER to combine the hypotheses. The disadvantage of ROVER is that it requires decoding to be carried out for each network, in addition to computing acoustic model scores for each network which is all that is required for score combination.

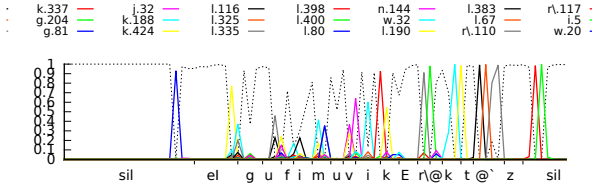
While we can train diverse CTC systems to estimate CD phone posteriors in a shared output space, with CTC the timing of the output symbols is arbitrary and we find that the timing of the spikes is different for different networks. Simple score fusion will not work with CTC since combining output posteriors by weighted averaging leads to meaningless scores where the strong signals from one network counteract the strong-but-differently-timed signals from another network. Temporal pooling or using constraints like those of Section 3.2 may be able to mitigate this disadvantage but we have not investigated further.

As an alternative we propose using the technique recently proposed for conventional speech recognition by Saon *et al.* [11]. They take two independently-trained networks and combine their final softmax layers by averaging together the contributions from each of the sub-networks. With further retraining to either cross-entropy or sequence-discriminative criteria, the joint network can be rebalanced to give performance superior to any of the component networks.

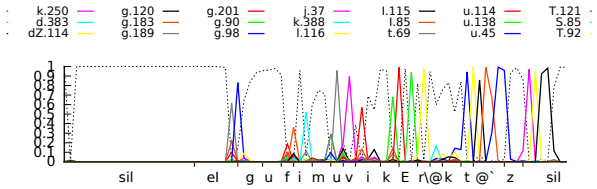
We argue that this technique has the potential to overcome the timing issues of score combination, since the joint retraining will force the networks to synchronize, while still only requiring a single decoding for the combination.

Experimental results are shown in Table 3. It can be seen that the combinations of CTC-models constructed in this way can outperform the original models, but after sequence training there is no improvement with respect to the best individual model. Similarly combining sequence-trained models does not bring significant improvement.

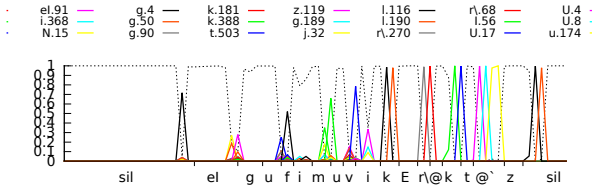
It can be seen that ROVER combination of 3 models did bring some improvement in WER (3% relative), but not as large as we have previously seen for conventional hybrid LSTMs, even for less-diversely constructed models.



(a) 8 frame fully-connected



(b) 5 frame convolution



(c) Combination

Fig. 5: Output timings for two independently CTC-trained networks and the joint network with a common softmax layer.

Model	Noisy set WER
Noisy model	14.5
Noisy model trained with noisy model's targets	20.9
Noisy model trained with noisy model's outputs	20.9
Noisy model trained with noisy model's targets during training	16.1
Clean model	21.4
Noisy model trained with clean model's targets	18.8
Noisy model trained with clean model's targets + retraining	15.4

Table 4: *Results from training directly with CTC or with knowledge transfer from one network to another.*

3.5. Knowledge transfer

In conventional training of hybrid neural network systems for speech recognition, it is common to train the network with a cross-entropy loss with respect to fixed targets which are determined by forced-alignment of a set of acoustic frames with a written transcript, transformed into the phonetic domain. Forced-alignment is the process of finding the maximum-likelihood label sequence for the acoustic frames and gives labels for every frame either in $\{0, 1\}$ for *Viterbi* alignment or in $[0, 1]$ for Baum-Welch alignment.

For GMMs, it is common practice to use the EM algorithm to iteratively improve a model by using it to align the data (E step) and then optimizing the parameters (M step). With DNNs, where every utterance is used in each of many epochs of training, it is common to store a fixed alignment from a previous "best" model and use it through many epochs of stochastic gradient descent, though it has been shown that continuous re-alignment is also feasible [2]

Here we explore the idea of using a variety of alternative alignment strategies in conjunction with the CTC algorithm. In the CTC algorithm, the current model is used to compute a target alignment in the form of the posteriors of the alignment (equivalent to the Baum-Welch alignment). These targets are used for a cross-entropy training, but are naturally recomputed with the latest model throughout training. We naturally wonder whether it is feasible to train a model to match fixed alignments computed with a previous "best" CTC model.

Alternatively, in the process of "distillation", Hinton *et al.* [12] have shown that it is possible to train a model to match the output distribution of an existing model. Here the new model is able to learn the "dark knowledge" stored in the original model and encoded in the distribution of outputs for a given input: where a Viterbi target would treat one label as correct and all others as incorrect, the output distribution of a trained network encodes the confusibility between classes. Thus as an alternative to training a network to match the targets computed by the Baum-Welch algorithm on its own outputs or those of another network, it is also feasible to train a network to match the output distribution of a network directly. The Baum-Welch algorithm has the advantage of employing the temporal constraints, but the "distillation" procedure has the advantage of transferring the "dark knowledge" from one net to another. Naturally all three methods of computing the targets can be employed, and we can optimize a weighted combination of the three losses. With the additional hyperparameter of the "temperature" of distillation, the option of using a conventional alignment cross-entropy loss from a separate output layer (as was found to improve stability and speed of convergence in [2]), the space of loss-functions becomes large, even without variation over time or considering their interaction with sequence-discriminative training. Here we describe some preliminary experiments to explore this space.

First we use an existing, pre-trained on noisy data to generate targets for a second network being trained from scratch. Targets are either taken directly from the first network's outputs, with no softening, or by applying the CTC algorithm to the first networks' outputs. Table 4 shows that transferring the targets directly with either method performs about the same, but does not achieve the same performance as training directly with the CTC algorithm.

Next, we jointly train two networks from scratch, where one is trained with CTC and for every utterance its targets are used to train the second network. This network achieves a better WER than training directly to the targets of a pre-trained network. We argue that this is because the network is not simply trying to match some optimal targets by a cross-entropy loss but is "relaxing into" a solution that has targets consistent with its own alignment. The CTC algorithm inherently needs to self-align to achieve optimal performance.

Training a new network on noisy data using the targets of a network already trained on aligned clean data allows the noisy-trained network to outperform the clean network on the noisy data, but even with further retraining, the network does not achieve the same performance as a network trained with CTC only on noisy data.

While there is a large space of possibilities for exploring knowledge transfer in CTC networks, including "distillation", and combining multiple objectives, our initial experiments lead us to believe that knowledge transfer with CTC is much harder than for conventional acoustic models.

4. SUMMARY & CONCLUSIONS

We have described a number of experiments with CD-CTC-sMBR LSTM RNNs. We have shown that these models can be successfully trained to recognize child and adult noisy speech. We have shown that these models are considerably faster in inference than CLDNN models achieving similar accuracy and that the latency in decoding can be reduced by constraining alignments during training. We have also shown that they can be trained with a convolutional input layer. We explore two strategies for model combination, showing improvements by fusing CTC-trained networks at the softmax layer, but finding improvements after sequence training only by ROVER combination of multiple models. Finally we have explored knowledge transfer between CTC networks and find that while knowledge can be transferred, this is not so simple as reusing alignments in conventional speech systems.

5. REFERENCES

[1] Alex Graves and Jürgen Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 12, pp. 5–6, 2005.

[2] H. Sak, A. Senior, K. Rao, Ozan Irsoy, A. Graves, F. Beaufays, and J. Schalkwyk, "Learning acoustic frame labeling for speech recognition with recurrent neural networks," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[3] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," in *To appear in Interspeech*, 2015.

[4] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd International Conference on Machine Learning*. ACM, 2006, pp. 369–376.

[5] A. Senior, H. Sak, and I. Shafran, "Context dependent phone models for LSTM RNN acoustic modelling," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[6] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[7] H. Liao, G. Pundak, O. Siohan, M.K. Carroll, N. Coccaro, Q.-M. Jiang, T.N. Sainath, A. Senior, F. Beaufays, and M. Bacchiani, "Large vocabulary automatic speech recognition for children," in *To appear in Interspeech*, 2015.

[8] T. N. Sainath, O. Vinyals, A. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[9] A. Robinson and F. Fallside, "A recurrent error propagation network speech recognition system," *Computer Speech and Language*, vol. 5, no. 3, pp. 259–274, 1991.

[10] J.G. Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)," in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, Santa Barbara, CA, Dec. 1997, pp. 347 – 354.

[11] G. Saon, H.-K. J. Kuo, S. Rennie, and M. Picheny, "The IBM 2015 english conversational telephone speech recognition system," *ArXiv e-prints*, , no. 1505.05899, 2015.

[12] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," *ArXiv e-prints*, , no. 1503.02531, Mar. 2015.