APPLICATION OF PROGRESSIVE NEURAL NETWORKS FOR MULTI-STREAM WFST COMBINATION IN ONE-PASS DECODING

Sirui Xu and Eric Fosler-Lussier

The Ohio State University
Department of Computer Science and Engineering

ABSTRACT

Many state-of-the-art automatic speech recognition (ASR) systems adopt system combination techniques to improve recognition performance. In this paper, we investigate the possibility of transferring knowledge between models for different noisy speech domains and integrating these models via system combination. The first contribution of our work is the use of progressive neural networks for modeling the acoustic features of noisy speech. We train progressive neural networks on subdivided noisy data to achieve knowledge transfer between different noise conditions. Our second contribution is an improved multi-stream WFST framework that combines the output of the progressive networks at longer timescales (e.g., word hypotheses). The score fusion is performed by a trained LSTM at the word boundary on the decoding lattice. By adopting both knowledge transfer and system combination techniques, we achieve improved performance compared with independently trained deep neural networks.

Index Terms— Progressive neural networks, system combination, automatic speech recognition

1. INTRODUCTION

The use of artificial neural networks in speech recognition has been studied for a couple of decades. The introduction of more efficient training methods has enabled neural networks to grow deeper with larger modeling powers. Since the spread of deep neural networks (DNNs), applications of neural network based speech recognition systems have generally outperformed traditional GMM systems.

System combination is one way to improve final speech recognition results. In this area of research, several kinds of combination techniques have been proposed, from frame-level acoustic score fusion, e.g. multi-band or multi-stream systems, to word-level combination that works on recognized sentences, e.g. ROVER [1] and CNC [2, 3]. Among these techniques, multi-band and multi-stream combination [4, 5] combines posterior scores generated by DNN acoustic models at the frame level and gains improved recognition performance. During the DNN training, datasets can be divided into sub-bands or subsets, and a DNN is trained independently from other DNN models on one of the sub-bands or subsets.

A potential issue of subdividing datasets is that each DNN model is trained with a smaller amount of data, which may result in a suboptimal modeling power in each sub DNN system. Inspired by transfer learning that can transfer learned knowledge from other datasets to the current models, we experiment with applying knowledge transfer techniques to system combination frameworks. Our proposed method treats sub DNN systems as correlated instead of independent entities, where each sub system, in addition to its corresponding sub dataset, also takes the output of other trained sub systems as input, which achieves knowledge transferring between sub systems.

Progressive neural networks (ProgNets) [6] can be utilized to transfer knowledge between domains. ProgNets were first applied in reinforcement learning on Atari tasks, where the visual features learned in one game were transfered to another game during the training, and it has shown to produce positive transfer between even very different games. In addition, ProgNets have the ability to prevent catastrophic forgetting by freezing the parameters of the previously trained model during training. ProgNets have proved to be effective in training robots and emotion recognition [7, 8].

In this paper, we report our experiments integrating ProgNets into system combination for robust speech recognition. We adapt our earlier work on multi-stream WFST recognition, integrating a neural network combination function. In Section 2, we introduce both the ProgNets and multi-stream WFST frameworks. Section 3 describes our datasets and experimental setup. Finally, we present and discuss our results and future work.

2. METHODS

2.1. Progressive Neural Networks (ProgNets)

ProgNets are constructed by multiple columns, each of which corresponds to a neural network. For each new task, a new column will be created based on previously trained system(s). During the training, all weights of previously trained systems are frozen and the output of each layer of the frozen networks are then fed as input of the corresponding layers in the new

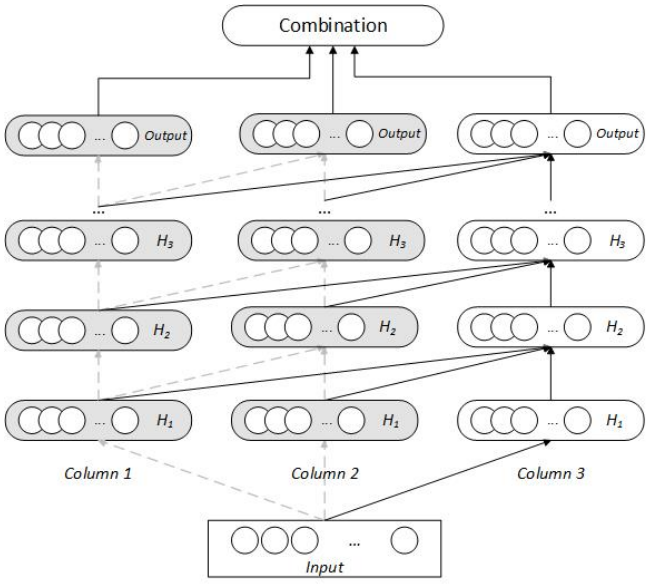

Fig. 1. Example of a progressive neural network. The first two columns are trained previously and connected through lateral connections to the current column.

model, through lateral connections.

A progressive network with a single column is just a standard deep neural network; as the number of columns increases, the input to a hidden layer becomes the activation of its lower layer and also that of the corresponding layers of previous columns. Each layer in a progressive network can be expressed as:

$$H_i^{(k)} = f\left(W_i^{(k)} H_{i-1}^{(k)} + \sum_{j<k} U_i^{(k:j)} H_{i-1}^{(j)}\right), \qquad (1)$$

where $W_i^{(k)}$ is the weight matrix of layer i of column k, $U_i^{(k:j)}$ is the weight matrix of the lateral connections between column k and column j, $H_i^{(k)}$ is the activation of layer i in column k, and f is the activation function. In [6], an adapter layer is used to transform a lateral connection when the structure between columns of ProgNets is different. However, we do not use adapter layers in our system, because we adopt a homogeneous structure for all the columns in the ProgNets.

Figure 1 shows the structure of a progressive neural network. The dashed lines are used for training the first two columns and are frozen for the current training. In the figure, the columns of the ProgNets are presented as having the same shape, but they do not have to be. ProgNets allow columns with different network structures, in which case adaptations and weight scalings may need to apply.

2.2. Multi-stream WFST combination framework

Different techniques have been proposed for system combination in speech recognition, including ROVER, CNC and multi-band and multi-stream systems. Although these methods have achieved improved performance, there lacks an effective framework that allows the combination of acoustic models with different decision tree structures at different levels of the speech recognition pipeline.

In our previous work [9], we proposed a WFST framework for integrating disparate systems while decoding in one pass. Rather than combining the word sequences output by different recognition systems, this framework tracks multiple scores from different sources during lattice generation, and combines scores at some intermediate level (frame, state, phone, word, or utterance). This allows the combination of different types of models in the speech recognition pipeline (e.g. acoustic models, pronunciation models and language models) in the decoding phase to achieve better single-pass recognition performance. We extend the standard one-label WFST to carry multiple labels and weights on the arcs to combine different models in decoding. When used for acoustic model combination, each of the labels corresponds to a tied triphone state from a system, and the weight is the acoustic score computed by the corresponding acoustic model.

The semiring structure in this framework is also extended for weight computation. A function will be used to decide how the scores from different models will be fused at the desired combination point. The most straightforward choice of the combination function is selecting the one with the maximum partial likelihood, but other functions, like averaging, inverse entropy or neural networks can be used. Figure 2 illustrates the multi-stream WFST framework. As shown in the figure, each arc has a vector of scores, since we extend the WFST to carry information from multiple models.

We use a novel approach by taking the original speech signal and deciding which streams to trust in building a combination function. We train an LSTM to predict whether a stream is likely to output the correct hypothesized unit (word). The input of the LSTM is a sequence of acoustic frames and the output consists multiple sigmoid units, each of which corresponds to an acoustic model. For each frame, the LSTM outputs a set of scores, which can be seen as the confidence of the systems and used to weight the posteriors. The training targets are generated by comparing the alignment from each subsystem with the alignment generated by the baseline system. The target of each frame for the corresponding sigmoid unit is 1 if the labels match, otherwise 0. Since LSTMs are able to accumulate information about the input over time, we expect that it can be better at predicting the most suitable acoustic models by capturing the noise information in longer range of the acoustic input.

In this study, we focus on combining acoustic models trained for disparate noisy environments. These acoustic

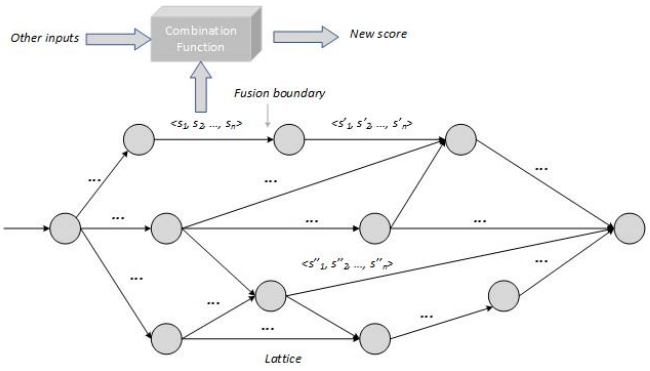

Fig. 2. Illustration of the multi-stream WFST framework. The combination is executed on a lattice.

models each have different senone decision trees, so combining information at the word level or frame level is easier than at the HMM state level. See [9] for additional model details.

3. DATASETS AND FEATURES

For our experiments, we used two noisy datasets: CHiME2 [10] and CHiME3 [11]. Both datasets are constructed based on the WSJ0 corpus.

The CHiME2 dataset includes 7138 utterances of 83 different speakers. For each of the utterances, a randomly selected SNR within the range of -6 to 9 dB is applied. In total there are 6 SNR levels in the training set, which are -6, -3, 0, 3, 6, 9 dB. The development set contains 409 noisy utterances from 10 speakers, and the evaluation set includes 330 noisy utterances from 8 other speakers. Our approach is to use a different progressive network for each SNR condition.

CHiME3 dataset contains simulated and real noise speech utterances, which are generated in 4 different noise environment: café (CAF), street junction (STR), public transport (BUS) and pedestrian area (PED). The training set has a total of 8728 noisy utterances, with 1600 real noisy utterances recorded by 4 speakers and 7138 simulated noisy utterances from the 83 recorded speakers of the WSJ0 SI-84 training set. The development set consists of 3280 utterances of 4 speakers other than the ones included in the training set, and the evaluation set contains a total of 2640 utterances of the 4 other speakers. For this dataset, we construct a progressive network for each noise condition.

The features used to train the DNNs are 13-dimensional MFCC features extended with deltas and double-deltas. Then they are adapted with LDA+MLLR transformations. Finally the fMLLR transformation is applied to reduce speaker variance. We note that other feature representations may produce better recognition results, but we use this setup to be consistent with prior work.

4. EXPERIMENTAL SETUP

We use fully connected DNNs as the building blocks for the experiment. The DNNs used as the baseline in both of the CHiME2 and CHiME3 experiments shared the same structure, which had 7 sigmoid layers and a softmax output layer. The baseline DNNs were trained on all of the noisy training data, and the state-level minimum Bayes risk (sMBR) criterion was used, before which the cross-entropy trained DNNs were used to generate the alignments as well as the lattices for the sequence training.

We then divided the noisy training into subsets, each of which corresponded to one of the noise conditions. For comparing with the ProgNets, separate DNNs, which shared the same structure of the baseline DNNs, were trained, and combined with the multi-stream WFST framework.

Each column in the ProgNets also used the same structure as the baseline systems. Each ProgNet was trained using data from a sub training set. We highlight the differences across datasets below:

CHiME2: in the first experiment, we trained 3-column ProgNets. In this configuration, the output column corresponds to the target SNR (and is trained on the corresponding SNR subset), and the other two columns are chosen to be the subsets which have the closest SNR levels to the target SNR. For example, when training ProgNets on the data subset with 0 dB SNR, the other 2 columns will be the ones corresponding to -3 dB and 3 dB SNR. We also trained a 6-column (full) ProgNet, the columns of which were progressively trained from -6 dB to 9 dB (except the target SNR), followed by an output column of the target SNR.

CHiME3: For each of the four different noise conditions, a full 4-column ProgNet was trained. The predecessor columns were taken from other noise conditions in random order.

For training the ProgNets, parameters of the previously trained columns were kept fixed and only the current column and the lateral connections were trainable. The dropout, with the rate of 0.5, was used during the training of the ProgNets to avoid overfitting.

For system combination, we experimented with 2 ways to combine the separately trained DNNs and ProgNets. We first tried the frame-level posterior fusion, which is one of the most straightforward ways to combine different acoustic models. We took the average of the posteriors output by the ProgNets and used the average score as the acoustic score in the standard WFST decoding process. Then we experimented with the word level combination, under the multi-stream WFST framework. Here, a 2-layer LSTM was trained to predict the most reliable models to be used at the word boundary.

5. RESULTS

In this section, we report our results obtained from the above experiments. Table 1 shows the word error rates for com-

Experiments	Combination level	WER
Baseline	-	21.1%
Independent DNNs	Frame-avg	20.9%
	Word-lstm	20.4%
ProgNets-3col	Frame-avg	20.7%
	Word-lstm	20.2%
ProgNets-full	Frame-avg	20.6%
	Word-lstm	20.2%

Table 1. Experiment results for the CHiME2 dataset. The row of ProgNets-3col shows the results for the ProgNets with 3 columns, and the row of ProgNets-full shows the results for the ProgNets with 6 columns correspnding to the 6 SNR levels in CHiME2.

	Baseline	ProgNets-full
-6 dB	36.9%	36.3%
-3 dB	27.3%	25.7%
0 dB	21.2%	20.1%
3 dB	16.1%	15.3%
6 dB	13.1%	12.7%
9 dB	11.9%	11.6%

Table 2. WER breakdown for CHiME2 test set according to the SNR levels for the baseline and full progressive nets combined using the word-based LSTM combination.

bining independent DNNs and ProgNets on CHiME2 and CHiME3 datasets.

For the CHiME2 experiments, combining independently trained DNNs and ProgNets both performed better than the single DNN baseline system, although the improvement of combining the posteriors of independently trained DNNs at the frame level is not significant. As Table 1 shows, for the frame-level combination, ProgNets outperformed independently trained DNNs by 0.2% and 0.3%. When the columns of ProgNets increased from 3 to full, the WER deduction was not significant. This may imply that useful information was carried in the data of the closest SNR levels.

Table 2 shows the breakdown of WERs for CHiME2 test set according to the SNR levels. As can be seen in the table, in general the WER deduction is higher in more noisy data than less noisy data. But for the -6 dB subset, the deduction is not as significant as that of the -3 dB. This possibly could be because the predecessor networks were all based on higher SNR data.

For the experiments on the CHiME3 dataset, the best performance was achieved by using 4-column ProgNets with the word-level combination. As shown in Table 3, compared with the DNN baseline trained with all noisy data, the WER deduction is 1.1% for real noisy test set, and 0.9% for simulated noisy test set. Similar to the CHiME2 experiments, ProgNets consistently outperformed independently trained DNNs in both frame-level posterior fusion and word-level combination.

Experiments	Noise type	WER	
Baseline	Real	18.9%	
	Simu	20.4%	
		Frame-avg	**Word-lstm**
Indept. DNNs	Real	18.6%	18.3%
	Simu	20.4%	20.2%
ProgNets-full	Real	18.4%	17.8%
	Simu	20.0%	19.5%

Table 3. Experiment results for the CHiME3 dataset. The row of ProgNets-full shows the results of the ProgNets with 4 columns corresponding to the 4 noise environments in CHiME3.

For both CHiME2 and CHiME3 experiments, the multi-stream WFST framework improves recognition performance by combining different acoustic models at the word level. We can see in the tables that the word-level combination outperformed the frame-level posterior fusion by at least 0.4% for the CHiME2 dataset and 0.2% for the CHiME3 dataset. The largest margin occurred in the real noisy test set, which is 0.6%. We expected the improvement at the word-level, because more frames were accumulated and more information about the noise conditions can be used for combination.

6. CONCLUSIONS

In this paper, we propose the use of progressive neural networks for multi-stream WFST combination in one-pass decoding. We used the noisy speech data from CHiME2 and CHiME3 datasets. To take advantage of the ability of ProgNets in transferring knowledge between different domains or datasets, we sub-divided the data according to different noise conditions so that the trained models can share the information about the noisy data. In addition, the word-level combination in the multi-stream WFST framework further helps improve the performance, as it can make use of a longer range of acoustic information for the combination. Combining the two techniques, we achieved a reasonable improvement over the baseline.

For future work, we would like to experiment with the ProgNets technique for different types of neural networks, such as RNNs and CNNs. In the experiments we reported in this paper, all the columns of the ProgNets shared the same neural network structure, but the state-of-the-art speech recognition systems combine different types of neural network acoustic models[12]. Since different types of models can capture different characteristics of data to allow the models to compensate for each other, it is worth exploring the possibility of using progressive neural networks or other knowledge transferring techniques for different model structures in noisy speech recognition and system combination.

7. REFERENCES

[1] Jonathan G Fiscus, "A post-processing system to yield reduced word error rates: Recognizer output voting error reduction (ROVER)," in *Proceedings IEEE Workshop on Automatic Speech Recognition and Understanding*. IEEE, 1997, pp. 347–354.

[2] Lidia Mangu, Eric Brill, and Andreas Stolcke, "Finding consensus in speech recognition: Word error minimization and other applications of confusion networks," *Computer Speech & Language*, vol. 14, no. 4, pp. 373–400, 2000.

[3] Gunnar Evermann and PC Woodland, "Posterior probability decoding, confidence estimation and system combination," in *Proc. Speech Transcription Workshop*. Baltimore, 2000, vol. 27, pp. 78–81.

[4] Hynek Hermansky, Sangita Tibrewala, and Misha Pavel, "Towards ASR on partially corrupted speech," in *ICSLP Proceedings Fourth International Conference on Spoken Language*. IEEE, 1996, vol. 1, pp. 462–465.

[5] Adam Janin, Dan Ellis, and Nelson Morgan, "Multi-stream speech recognition: Ready for prime time?," in *Proceedings Eurospeech*, 1999, pp. 591–594.

[6] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell, "Progressive neural networks," *arXiv:1606.04671*, 2016.

[7] Andrei A Rusu, Matej Vecerik, Thomas Rothörl, Nicolas Heess, Razvan Pascanu, and Raia Hadsell, "Sim-to-real robot learning from pixels with progressive nets," *arXiv:1610.04286*, 2016.

[8] John Gideon, Soheil Khorram, Zakaria Aldeneh, Dimitrios Dimitriadis, and Emily Mower Provost, "Progressive neural networks for transfer learning in emotion recognition," *Proceedings Interspeech*, pp. 1098–1102, 2017.

[9] Sirui Xu and Eric Fosler-Lussier, "A WFST framework for single-pass multi-stream decoding," in *Interspeech*, 2016, pp. 1908–1912.

[10] Emmanuel Vincent, Jon Barker, Shinji Watanabe, Jonathan Le Roux, Francesco Nesta, and Marco Matassoni, "The second CHiME speech separation and recognition challenge: Datasets, tasks and baselines," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2013, pp. 126–130.

[11] Jon Barker, Ricard Marxer, Emmanuel Vincent, and Shinji Watanabe, "The third CHiME speech separation and recognition challenge: Dataset, task and baselines," in *Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 504–511.

[12] Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig, "The Microsoft 2017 conversational speech recognition system," in *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5255–5259.