Autoregressive product of multi-frame predictions can improve the accuracy of hybrid models

Navdeep Jaitly[1], *Vincent Vanhoucke*[2], *Geoffrey Hinton*[1,2]

[1]University of Toronto
[2]Google Inc.

ndjaitly@cs.toronto.edu, hinton@cs.toronto.edu

Abstract

We describe a simple but effective way of using multi-frame targets to improve the accuracy of Artificial Neural Network-Hidden Markov Model (ANN-HMM) hybrid systems. In this approach a Deep Neural Network (DNN) is trained to predict the forced-alignment state of multiple frames using a separate softmax unit for each of the frames. This is in contrast to the usual method of training a DNN to predict only the state of the central frame. By itself this is not sufficient to improve accuracy of the system significantly. However, if we average the predictions for each frame - from the different contexts it is associated with - we achieve state of the art results on TIMIT using a fully connected Deep Neural Network without convolutional architectures or dropout training. On a 14 hour subset of Wall Street Journal (WSJ) using a context dependent DNN-HMM system it leads to a relative improvement of 6.4% on the dev set (*test-dev93*) and 9.3% on test set (*test-eval92*).

Index Terms: product models, DNN-HMM, ANN-HMM

1. Introduction

The use of forced alignments from Gaussian Mixture Model - Hidden Markov Models (GMM-HMM) for training neural networks suffers from several drawbacks. Firstly, the quality of the GMM-HMM system itself affects the quality of alignments generated. GMM-HMM systems make strong assumptions about the data generation process, such as independence of acoustic frames given states, that are not quite true. As a result the GMM-HMM model suffers from weird artifacts [1]. In figure 1 we present results of an experiment that shows that forced alignments may not provide the best data to train neural networks with. For this, we used the forced alignments from a simple GMM-HMM system and corrupted the boundaries between phone internal states. To be more specific, we generated forced alignments from a tri-state monophone GMM-HMM system trained on TIMIT. For each segment corresponding to a phoneme we re-segmented the internal state boundaries by distributing them equally within the three internal states. Thus each segment between the start frame and the end frame assigned to a phoneme was split into three subsegments, and these were assigned the start state, the middle state and the end state of the triphone HMM. The effect of this is to generate an alignment that is smoothed out. Note that the amount of data presented to the DNN models is the same in both cases - the procedure does not augment that training data with different, fuzzy examples each iteration and the smoothing is done only at the start of the training. So it should not lead to better DNN models through regularization. Figure 1 shows the results of training models of different depths using the smoothed alignments for TIMIT. Clearly, the phone recognition accuracy of the models trained with the corrupted boundaries is as good as, if not actually *better* than that the models trained with the forced alignments. The reason for these potential improvements is beyond the scope of this paper, but might be explained by more robust estimates of rare states or better matching the equal transition probability of the HMM.

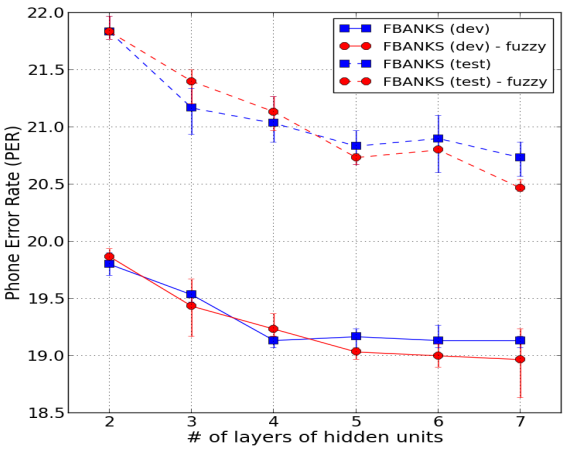

Figure 1: *Phone and Word Error rates (PER, WER) obtained with models trained on correct and smoothed alignments*

.

It has been shown that discriminative training of GMM-HMMs leads to qualitatively different type of forced alignments that can lead to improved phone accuracy without improvding the frame classification error [2]. Gillick [2] suggests that this must mean that discriminative training's (they used MPE) 'benefit only appears across sequences of frames'. It is natural to ask if neural networks trained even on forced alignments (not just in discriminative training) could leverage some sequential information.

Another main drawback of this neural network training for ANN-HMM systems is that each data case only provides $\log_2 M$ bits of information through the state labels (where M is the number of distinct states). While this drawback is shared by all classification algorithms, speech is a very structured modality and this structure is ignored in the neural network training. Recurrent neural network methods trained on forced alignments (such as [3, 4]) do not suffer from this problem since the entire sequence of targets is trained together and thus structure outputs are implicitly modelled by these methods.

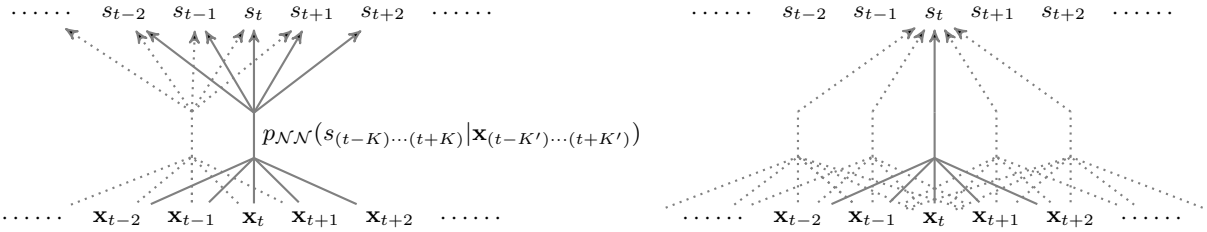

Figure 2: *Auto-regressive product model for speech recognition.* **Left:** *During training the neural network uses an input context of K' at each time point t to predict all the states within a context of K around t. In this example $K, K' = 2$.* **Right:** *During decoding the autoregressive scores for time point t from all contexts is averaged.*

In this paper we present a method that attempts to incorporate these insights into neural network training from forced alignments. We train a neural network to predict the phone states of all the frames within a context window of a central frame using the acoustic data around the same central frame with the same (or larger) context window as input. At test time we take a geometric average (product model) of the predictions for each frame from all the acoustic contexts that model the state of that frame in their output layer. Using this method we achieve state of the art results on TIMIT using a vanilla fully connected DNN - without any special techniques such as Convolutional neural network architectures or dropout training [5, 6, 7]. On a 14 hour subset of WSJ using a context dependent DNN-HMM system it leads to a relative improvement of 6.4% on the dev set (*test-dev93*) and 9.3% on test set (*test-eval92*).

Our approach to predicting multiple frames of outputs is hardly the first one. Recently Vanhoucke [8] trained a DNN to predict the output labels for multiple frames. However, no averaging of predictions was performed - instead the method was used to speed up decoding by performing a forward pass through the DNN's at a slower frame rate. Product models are not new to the hybrid speech recognition community either. Sim [9] shows an interesting application of product models to cross-lingual phone recognition by training a phone recognizer for English phones using a product model of ANN-HMM recognizers in Czech, Hungarian and Russian. Our approach is different from this in that we form a product from the same model applied convolutionally. In addition, at least for this paper, we do not learn the weights of the distributions making up the product and simply use the geometric mean of the distributions.

2. Methods

Here we describe the training of the neural networks and the decoding. Figure 2 presents an overview of the method.

2.1. Multi-frame Target Training

We train a neural network to predict the phone labels for multiple output frames given the acoustic data. The inputs to the neural networks are $2K' + 1$ frames of acoustic vectors $\mathbf{x}_{t-K'} \cdots \mathbf{x}_{t+K'}$ and the targets are the $2K + 1$ one-hot encoded phone labels $\mathbf{s}_{t-K} \cdots \mathbf{s}_{t+K}$ associated with K context frames around the center frame at time t[1]. The output layer is a set of $2K + 1$ independent softmax units, each with M phoneme label classes. Thus the conditional distribution of the output labels around time t is given by:

$$p\left(\mathbf{s}_{(t-K)\cdots(t+K)}|\mathbf{x}_{(t-K')\cdots(t+K')}\right) = \prod_{t'=t-K}^{t+K} p_{(t'-t)}(\mathbf{s}_{t'}|\mathbf{x}_{(t-K')\cdots(t+K')}) \quad (1)$$

where $p_{(t'-t)}(\mathbf{s}_{t'}|\mathbf{x}_{(t-K')\cdots(t+K')})$ is the softmax associated with a delay of $(t' - t)$ frames from the central frame.

The gradients with respect to the parameters of the neural network can be computed by backpropagating the the softmax gradients at the the output layer. Learning is done using stochastic gradient descent (more details can be found in section 3).

2.2. Decoding with Autoregressive Targets (DART)

Once a model been trained to perform multi-frame target prediction all the predictions for the states of an individual frame can be combined together. Specifically, we take a geometric average of all the predictions for each time point t:

$$p(\mathbf{s}_t|\mathbf{X}) \propto \prod_{t'=t-K}^{t+K} p_{t-t'}^{1/K}(\mathbf{s}_t|\mathbf{x}_{(t'-K')\cdots(t'+K')}) \quad (2)$$

This is easily accomplished by averaging the activations of the softmax units (i.e. the logit values) associated with time t from the application of the neural networks to all time points $t-K$ to $t+K$. Extra inputs are appended to the start and end of the utterances for boundary cases - we simply repeat the input at the first frame for prefix frames and the last frame for suffix frames. DART does not need to average over all K frames but experiments suggest that using all the frames leads to better results (see table 1).

Instead of product averaging we could use a mean of all the predictions at each time point t, i.e.:

$$p(\mathbf{s}_t|\mathbf{X}) = \frac{1}{K} \sum_{t'=t-K}^{t+K} p_{t-t'}(\mathbf{s}_t|\mathbf{x}_{(t'-K')\cdots(t'+K')}) \quad (3)$$

3. Experiments and Discussion

We used TIMIT [10] and the 14 hour subset of WSJ [11] (*si-84*) for this paper. For TIMIT a bi-phone language model trained on TIMIT training utterances was used. For WSJ we used the *big dictionary* setup in Kaldi that adds common pronounciation variants to the default dictionary used for WSJ experiments (cmudict). A trigram language model was trained on the corpus

[1] We use bold-face $\mathbf{s}_t$ for one-hot encoded labels and s_t for the class label that it is assigned to.

provided with the WSJ CD. More details can be found in the *s5* recipe - model tri4b - of Kaldi which we used for this [12].

For the DNN-HMM models we used 15 frames of 40 dimensional filterbanks with deltas and accelerations as inputs. All the neural networks using the same inputs were pretrained with the same Deep Belief Network [13, 14]. We trained the deep neural networks using the annealing schedule and learning rates described in [15] with a small but critical change - the learning rates for the bottom two layers were reduced to 0.005 and 0.02 respectively. For TIMIT we used DNNs with sevel hidden layers of sigmoid units; for WSJ we used six layers as a compromise between depth and computational time. The TIMIT recipe had 180 output labels so the output layer was a set of $2K+1$ softmax units of 180 dimensions each - leading to output dimensions of $(2K+1)*180$. For WSJ we had 3385 states so the output layer was a set of $(2K+1)$ softmax units of 3385 dimensions each. Because of the large output dimensions, training was slower. Note that even though we average multiple predictions there is only one DNN system, so the number of parameters is almost the same as the number of parameters in a vanilla DNN-HMM system. The only difference is the extra parameters at the output layer, where we have as many softmax distributions as we have number of target frames, instead of a single softmax distribution. When $K=0$, i.e. we only predict the central state, these two are equivalent.

3.1. Effect of context window sizes

Table 1 shows the Phone Error Rate (PER) achieved on TIMIT using DNNs with five hidden layers of sigmoid units trained to predict different context sizes of outputs. We trained four sets of triplicate models to predict the labels of 1, 3, 7 and 15 frames respectively (i.e. $K=0,1,3,7$) using 15 frames of input (i.e. $K'=7$)[2]. For each model we averaged over different context window sizes to explore the impact of averaging. It can be seen that for each of these models averaging over multiple contexts improved the results. Also it can be seen that larger output window sizes lead to impoved results (larger values of K, K' did not appear to produce significant gains). Interestingly, all models had comparable accuracy when they used the same DART context size for averaging. This leads us to conclude that gains are not achieved by some regularization effect, but instead because of model averaging.

3.2. Frame Error Rate of Models

We found that in addition to improved phone recognition results, the models also achieved much better frame error rates (FER) and log probabilities (which were computed using the geometrically averaged probability distributions). Figure 3 shows a comparison of the FER for three different runs of multi-frame DNN training (K=7) and the FER for three different runs of vanilla DNN training (K=0). Clearly, the multi-frame strategy leads to much better FER.

3.3. Impact of Depth of DNNs

Using $K, K'=7$ we then trained three DNN's of depths two to seven hidden layers on TIMIT. Figure 4 shows the PER achieved for different depths and compares it to the baseline with $K=0$. It can be seen that DART leads to a significant gain in accuracy over the baselines for both the dev set and the test set.

[2] $K=0$ is the baseline representing the vanilla DNN training for hybrid models.

Table 1: *DART results on TIMIT from five layer DNNs trained on different output context window sizes. For each context window size we decoded using different window sizes of autoregressive output targets. In each case 15 frames of acoustic inputs (i.e. $K'=7$) was used; each hidden layer had 2000 sigmoid units. Results are averages over three runs.*

training context (K)	DART context	PER	
		dev	test
0	0	19.2	20.8
1	0	18.9	20.6
	1	18.9	20.3
3	0	18.5	20.4
	1	18.3	20.3
	3	18.0	20.0
7	0	18.7	20.8
	1	18.5	20.5
	3	18.0	20.0
	7	17.6	19.5

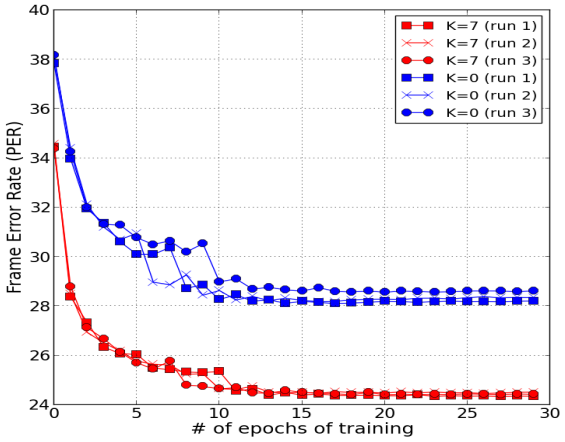

Figure 3: *Frame Error Rate (FER) of DART on TIMIT using DNNs with five hidden layers of 2000 units each. Each model used +/-7 frames of context. The DNN's were trained to predict +/-7 frames of context for the multi-frame models ($K=7$) and only the central frame for the non-multi-frame models ($K=0$).*

Note that we used fully connected deep neural network (DNN) models for this and achieved accuracy significantly better than those reported for simple Convolutional Neural Network - Deep Neural Network (CNN-DNN) - HMM systems ([5, 6]) and comparable to carefully crafted CNN-DNN-HMM model with heterogeneous pooling in [16] that was trained with dropout. It is our expectation that the gains are complementary, and similar gains would be produced when these ideas are applied to convolutional and other discriminative models.

3.4. WSJ Results

For WSJ, $K, K'=7$ was used. Figure 5 shows the FER for triplicate runs of multi-frame target models and compares it to the FER for three runs of single target models. Clearly a much better FER is achieved once more.

Table 2 shows a numerical summary of the results. Since, $K=7$ was used for these models, the output dimensionality was 3385x15. It can be seen that for *WSJ-si84* a relative im-

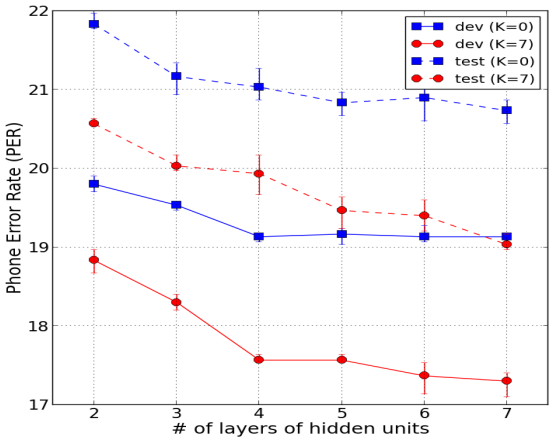

Figure 4: *PER of DART on TIMIT using DNNs of different depths. Each model used +/-7 acoustic frames of context ($K' = 7$). The DNN's were trained to predict +/-7 frames of context for the multi-frame models ($K = 7$) and only the central frame for the regular models ($K = 0$).*

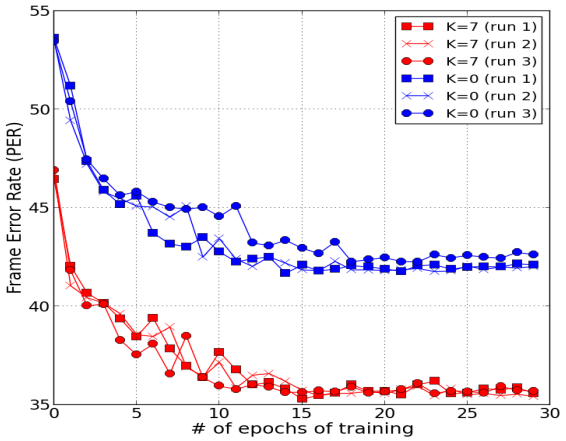

Figure 5: *Frame Error Rate (FER) of DART on WSJ using DNNs with six hidden layers of 2000 units each. Each model used +/-7 frames of context. The DNN's were trained to predict +/-7 frames of context for the multi-frame models ($K = 7$)and only the central frame for the non-multi-frame models ($K = 0$).*

provement of 6.4% was seen on the development set *test-dev93* and a relative improvement of 9.3% was seen on the test set *test-eval92*, over the baseline system. Further improvements may be possible by experimenting with the parameters of the decoder but we did not explore this avenue.

Table 2: *DART results using context window of +/-7 in input data and output states. Results are averages over three runs.*

dataset	architecture	product	PER / WER	
			dev	test
TIMIT	$(2K)^7 - 180$	N	19.1	20.9
	$(2K)^7 - (180x15)$	Y	**17.3**	**19.0**
WSJ-si84	$(2K)^6 - 3385$	N	9.4	5.4
	$(2K)^6 - (3385x15)$	Y	**8.8**	**4.9**

3.5. Geometric Averaging Compared to Arithmetic Averaging

We decoded each of the models trained on TIMIT in section 3.3 using geometric and arithmetic averaging. Table 3 shows a comparison of the average PER over three runs. It is clear that geometric averaging consistently outperforms arithmetic averaging here. Also, the trend with depth is much more consistent for geometric averaging. A possible explanation for why geometric averaging outperforms arithmetic averaging is that geometric averaging acts like constraints - solutions that violate any one of the predictions sharply are discouraged under this model. Arithmetic averaging, on the other hand leads, accepts solutions as long as one of the models is quite happy with the solution; thus it is susceptible to bad decision boundaries of models that have been overfit significantly.

4. Conclusions

We have shown that using an autoregressive product of a DNN-HMM system trained to predict the phone labels of multiple frames can improve speech recognition accuracy. The autoregressive model bears a resemblance to RNN's because it attempts to predict states over a range of frames. These connections need to be further explored. In this paper the predictions at multiple time points were trained independently and a simple geometric average was used at test time. Model combination approaches frequently benefit by using weighted combinations. In the future we will explore these avenues further. Lastly, it is interesting to note that geometric averaging outperforms arithmetic averaging here; it will be interesting to see if this observation can be applied to training ensembles of models for speech recognition in new ways.

While we trained these models on forced alignments, it is likely that sequential training methods (such as [17]) could also benefit from this approach.

Table 3: *A comparison of the geometric average to the arithmetic average of the autoregressive distributions on TIMIT test set using models of different depths. Results are averages over three runs.*

averaging type	# of layers of hidden units					
	2	3	4	5	6	7
geometric	20.6	20.0	19.9	19.5	19.4	19.0
arithmetic	20.8	20.3	20.4	19.9	19.6	19.4

5. Acknowledgements

We would like to express our gratitute to Nitish Srivastav for valuable discussions. This research was funded by a Google Research Award to Geoffrey E. Hinton

6. References

[1] D. Gillick, L. Gillick, and S. Wegmann, "Don't multiply lightly: Quantifying problems with the acoustic model assumptions in speech recognition," in *Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on*, Dec 2011, pp. 71–76.

[2] D. Gillick, S. Wegmann, and L. Gillick, "Discriminative training for speech recognition is compensating for statistical dependence in the hmm framework," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, March 2012, pp. 4745–4748.

[3] A. Graves, N. Jaitly, and A. Mohamed, "Hybrid speech recognition with deep bidirectional lstm," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, Dec 2013, pp. 273–278.

[4] A. Robinson, "An application of recurrent nets to phone probability estimation," *Neural Networks, IEEE Transactions on*, vol. 5, no. 2, pp. 298–305, Mar 1994.

[5] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition," 2013.

[6] O. Abdel-Hamid, A.-r. Mohamed, H. Jiang, and G. Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.

[7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *CoRR*, vol. abs/1207.0580, 2012.

[8] V. Vanhoucke, M. Devin, and G. Heigold, "Multiframe deep neural networks for acoustic modeling," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, May 2013, pp. 7582–7585.

[9] K. C. Sim, "Discriminative product-of-expert acoustic mapping for cross-lingual phone recognition," in *Automatic Speech Recognition & Understanding, 2009. ASRU 2009. IEEE Workshop on*. IEEE, 2009, pp. 546–551.

[10] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT acoustic phonetic continuous speech corpus CDROM," 1993. [Online]. Available: http://www.ldc.upenn.edu/Catalog/LDC93S1.html

[11] D. B. Paul and J. M. Baker, "The design for the wall street journal-based csr corpus," in *DARPA Speech and Language Workshop*. Morgan Kaufmann Publishers, 1992.

[12] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011.

[13] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[14] G. E. Hinton and R. R. Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," *Science*, vol. 313, no. 5786, pp. 504–507, Jul. 2006.

[15] N. Jaitly and G. E. Hinton, "Using an autoencoder with deformable templates to discover features for automated speech recognition." in *INTERSPEECH*, F. Bimbot, C. Cerisara, C. Fougeron, G. Gravier, L. Lamel, F. Pellegrino, and P. Perrier, Eds. ISCA, 2013, pp. 1737–1740. [Online]. Available: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2013.html#JaitlyH13

[16] L. Deng, O. Abdel-Hamid, and D. Yu, "A deep convolutional neural network using heterogeneous pooling for trading acoustic invariance with phonetic confusion," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6669–6673.

[17] B. Kingsbury, "Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 3761–3764.