INTERSPEECH 2016
September 8–12, 2016, San Francisco, USA

Distilling knowledge from ensembles of neural networks for speech recognition

Yevgen Chebotar[1], *Austin Waters*[2]

[1]University of Southern California, Los Angeles, CA, USA
[2]Google Inc., New York, NY, USA

ychebota@usc.edu, austinwaters@google.com

Abstract

Speech recognition systems that combine multiple types of acoustic models have been shown to outperform single-model systems. However, such systems can be complex to implement and too resource-intensive to use in production. This paper describes how to use *knowledge distillation* to combine acoustic models in a way that has the best of many worlds: It improves recognition accuracy significantly, can be implemented with standard training tools, and requires no additional complexity during recognition. First, we identify a simple but particularly strong type of ensemble: a late combination of recurrent neural networks with different architectures *and* training objectives. To harness such an ensemble, we use a variant of standard cross-entropy training to distill it into a single model and then discriminatively fine-tune the result. An evaluation on 2,000-hour large vocabulary tasks in 5 languages shows that the distilled models provide up to 8.9% relative WER improvement over conventionally-trained baselines with an identical number of parameters.

Index Terms: acoustic modeling, knowledge distillation, ensembles, deep neural networks, long short-term memory

1. Introduction

A variety of work in speech recognition combines predictions from multiple models or systems to improve recognition accuracy. The ROVER [1] method, for example, tries to minimize errors by combining the outputs of parallel recognizers through a voting scheme. More recent work has explored combination at the acoustic model level [2], in particular by fusing together neural networks with distinct, complementary architectures. Examples include "stacking" posterior predictions from a deep neural network (DNN), convolutional neural network (CNN), recurrent neural network (RNN) [3] and hybrid architectures that fuse CNN and DNN structures earlier in the model [4, 5]. These approaches capitalize on each architecture's strengths to improve accuracy. While such methods can give significant improvements in recognition accuracy, these types of models can be difficult to deploy. First, typically both training and prediction infrastructure need to be modified in complex ways to support a new model architecture. Furthermore, the combined models may be too large or resource intensive for real-time use. Scoring with the stacked ensemble of [3], for instance, essentially requires making predictions for three separate models.

We use *knowledge distillation* or model compression [6, 7] to address such issues. At a high level, distillation involves training a new model, which we dub the *student*, to match the soft outputs (posteriors or logits) of an already-trained *teacher*, rather than hard class labels. We use Kullback-Leibler (KL) divergence to minimize the difference between student and teacher output distributions, which is similar to approach for performing model adaptation as described in [8]. In previous work, knowledge distillation was predominantly used for training small models using outputs of single large models. Romero et al [9] train thin deep neural networks to imitate outputs of large wide networks. Li et al [10] distill knowledge from a large DNN to a smaller DNN by matching their output distributions with KL-divergence. The authors use a large set of untranscribed data to create training labels for their student model and show significant performance improvement compared to the conventional training methods. Similarly, Chan et al [11] distill knowledge from a RNN to a DNN, which likewise improves the WER of the student model. In contrast, in our work, we focus on applying this technique to ensembles and show that this substantially increases the single model performance. In particular, each teacher is an ensemble of different recurrent neural networks and each student is a single such model. After distillation, the student model typically retains most of the predictive accuracy of its teacher. Crucially, while the former is large and slow, the latter is small and fast enough to be used in a production system. Furthermore, the distillation process can be carried out with standard training tools as it is equivalent to cross-entropy training with the ensemble posteriors as targets.

Producing strong student models with distillation first requires constructing teacher ensembles that have high predictive accuracy. The ensembles in this work are built simply by averaging the context-dependent state posteriors of individually-trained neural network models. We not only combine models with complementary architectures (in the spirit of [3, 4, 5]) but also different individual training criteria. In particular, we use combinations of Long Short-Term Memory (LSTM) [12] and Convolutional LSTM Deep Neural Networks (CLDNN) [13] models trained with both cross-entropy (CE) and state-level minimum Bayes risk (sMBR) criteria [14].

We evaluate both the ensembles and the student models distilled from them on 2,000-hour LVCSR tasks in 5 languages. Empirically, our ensembles capture the complementary properties of their constituent models and give significant improvements over any one individually. More importantly, the (much smaller) student models retain most of the predictive power of the ensembles, and sometimes exceed it when fine-tuned with discriminative sequence training. Overall, our training procedure provides up to 8.9% relative WER improvement over baseline models with an identical number of parameters.

The rest of this paper is structured as follows. Section 2 describes our particular ensemble method and an oracle-based metric for choosing models that will combine well. Section 3 describes how we use the ensembles to train and fine-tune the student models. Section 4 provides experimental results on LVCSR tasks to validate the proposed method. Finally, Section 5 concludes the paper.

2. Ensembles of acoustic models

This section explains how ensembles of LSTM and CLDNN acoustic models can be used to improve speech recognition performance. We describe how the combination is performed and how to choose strong ensemble candidates.

2.1. Combination of acoustic model scores

We use a simple but effective late combination method to construct acoustic model ensembles. Given N models that have already been trained on available data (through whatever means), we combine their frame-level predictions by taking a weighted average of their context-dependent (CD) state posteriors. That is, for each frame of audio x, the ensemble emits a vector of posteriors over CD states s computed as

$$\mathbf{P}_{\text{ens}}(s|x) = \textstyle\sum_{i=1}^{N} w_i \mathbf{P}_{\mathbf{i}}(s|x), \tag{1}$$

where $\mathbf{P}_i$ are posteriors from the i-th model, $w_i \in [0, 1]$ is its weight, and $\sum_{i=1}^{N} w_i = 1$. This method does require that the constituent models have identical CD state inventories (i.e. corresponding outputs of the same dimensionality) and emit time-synchronous predictions. The conditions can be satisfied by construction when building the individual models.

To find the optimal weights, we perform grid search over the weight combinations and choose the one that leads to the best performance of the complete recognition system on a given data set.

2.2. Choosing a good ensemble

What types of models yield the best improvements when combined into an ensemble? As noticed in prior work (e.g. [15, 16]), models that tend to make errors on different inputs can compensate for one another in combination and make fewer errors than any model individually. One widely used ensemble method, boosting, induces such behavior by construction, by training a sequence of classifiers that attempt to correct the errors of the previous ones (e.g in [17]). In this work, as in [3, 4, 5], we instead use the fact that different neural network architectures can make systemically different errors. Unlike previous work, we also exploit the idea that different *training criteria* can lead to different errors.

In practice, it is desirable to have a quantitative measure that indicates which acoustic model combinations might lead to the highest gains in an ensemble. Such a metric seems difficult to compute because Equation (1) defines how predictions are combined at the frame level, while the ultimate goal is to minimize the word error rate. However, it is possible to bridge that gap by replacing the real combination function with an oracle that approximates the best possible ASR outcome of the ensemble. Specifically, the oracle

$$\mathbf{P}_{\text{oracle}}(s|x) = \textstyle\sum_{i=1}^{N} [\mathcal{O}(u) = i] \mathbf{P}_i(s|x) = \mathbf{P}_{\mathcal{O}(u)}(s|x), \tag{2}$$

where, in place of soft combination weights, the oracle $\mathcal{O}(u) \in 1 \ldots N$ gives all weight to the model that yields the fewest word errors on the current utterance u. Such an oracle is easy to construct for any set of transcribed utterances; building it just requires plugging each individual model into the ASR system and computing the number of word errors made on each utterance. As we show in Section 4.1, the real weighted-average ensembles never achieve the oracle performance, but the oracle's WER reliably predicts which combinations will perform best.

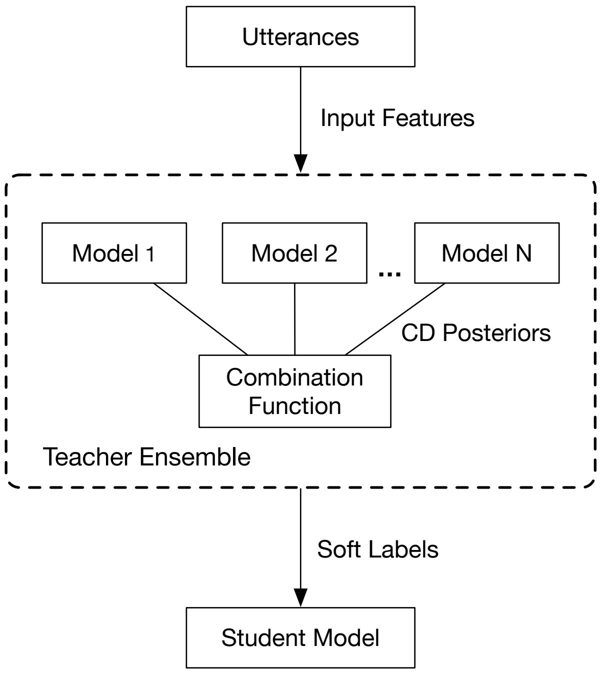

Figure 1: Knowledge distillation framework. Utterance input features are used to create soft labels from the teacher ensemble. CD posteriors of ensemble models are aggregated by using a combination function (e.g. weighted average), which provides training data for the student model.

3. Distilling knowledge from ensembles

The main idea of knowledge distillation [7] is to train a model (student) on outputs of another model (teacher). Given a data set with hard labels, i.e. where each data sample is assigned to a single label, we first train a teacher model that outputs probability distribution over all labels (soft labels) given the input features. If we have a well-performing large teacher model, it is possible to achieve better performance in a smaller student model by training it on the soft outputs of the teacher model instead of directly on the hard labels. Such training makes the student model mimic the behavior of the teacher model. The soft labels might contain more information about the underlying label distribution than the hard assignments and might be easier to learn for the student model.

In this work, each teacher is an ensemble of individually-trained models whose predictions are combined with Equation (1), and each student is a single CLDNN model. In practice the teacher ensembles perform substantially better than single models, but are too inefficient to use in real systems. The goal of distillation is to transfer the predictive accuracy of those ensembles into the single student models, which are efficient enough to deploy. In the following, we describe the approach of training the student based on the outputs of a teacher ensemble, as well as ways to further improve the student performance. Figure 1 shows our general framework for knowledge distillation. Given the input features from utterances, CD output posteriors of ensemble models are combined to create soft labels for training the student model.

3.1. Distillation through cross-entropy training

In our setting, the outputs of each teacher and student models are posterior probabilities over the same set of context-dependent states. In order to make a student model mimic its teacher, we train the former to emit posteriors that are as close as possible to those of the latter, given the same audio. This can be achieved by minimizing the Kullback-Leibler (KL) divergence [18] between student and teacher distributions. Letting $P_{\text{ens}}(s|x)$ be the posterior distribution of the teacher ensemble (defined in (1)), and $Q(s|x)$ the posterior distribution of the student model, we wish to minimize

$$\begin{aligned} D_{KL}(P_{\text{ens}}(s|x)\|Q(s|x)) &= \sum_i P_{\text{ens}}(s_i|x)\ln\frac{P_{\text{ens}}(s_i|x)}{Q(s_i|x)} \\ &= H(P_{\text{ens}},Q) - H(P_{\text{ens}}), \end{aligned}$$

with respect to the parameters of $Q(\cdot)$. Here,

$$H(P_{\text{ens}},Q) = \sum_i -P_{\text{ens}}(s_i|x)\ln Q(s_i|x)$$

is the cross-entropy (CE) error between the teacher and student distributions, and

$$H(P_{\text{ens}}) = \sum_i P_{\text{ens}}(s_i|x)\ln P_{\text{ens}}(s_i|x)$$

is the entropy of the teacher distribution. The teacher entropy is not a function of Q and thus has no influence on training; it vanishes when computing gradients with respect to the student model parameters. Thus, the KL-divergence is minimized by minimizing the CE error, and the optimization procedure becomes equivalent to standard cross-entropy training. The only difference is that instead of the one-hot CD states typically used as training labels, we substitute a soft distribution over the CD states from the teacher ensemble. This procedure is essentially a simplified version of the more general temperature-based distillation of Hinton et al. [7].

3.2. Improving student model

After running CE training on the soft outputs of a teacher ensemble, the student performance may be considerably lower than that of the teacher. We found that the following techniques can improve the student.

First, rather than initializing the student with random weights, we start it with a copy of the parameters from one of the ensemble models. In practice this causes the student to converge faster and have better performance. This scheme requires initializing from a model with the same architecture, e.g. if the student is a CLDNN model we have to use a CLDNN with analogous structure from the teacher ensemble. If there are several models with the suitable architecture available in an ensemble, we choose the model with the highest combination weight.

Second, further performance gains can be achieved by running discriminative sequence training on the student model after distillation. For this, we make a copy of the distilled student model and apply the sMBR training method of [19] on it. Doing so typically improves the student model, as we show in Section 4.2. sMBR training does not use the teacher ensemble; its training labels come from the original audio transcriptions.

4. Experiments

This section presents experiments to evaluate our methods for ensemble selection and knowledge distillation in multiple languages. All models described below are trained on sets of

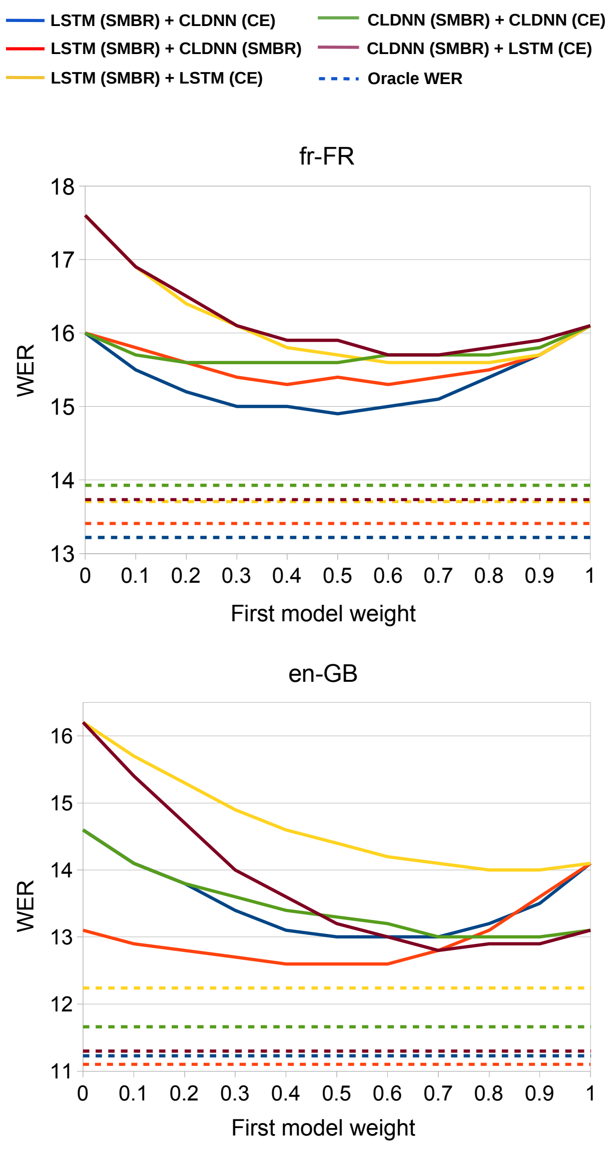

Figure 2: Word error rates of two-model combinations as a function of combination weight. x-axis shows the weight w_1 of the first model in the ensemble, the weight of the second model is $w_2 = 1.0 - w_1$. Dotted lines show WERs of oracles that choose the best model for each utterance. In both languages, the relative order of oracles indicates which model combinations work best. Sometimes (top figure) the best ensembles are those combining different architectures and training methods.

3 million utterances (approximately 2,000 hours) using asynchronous stochastic gradient descent optimization [20]. For evaluation purposes, we measure word error rates (WERs) on test sets of 30,000 utterances (approximately 20 hours) per language. All data sets are anonymized and hand-transcribed. The training sets are representative of Google's overall speech traffic, and the test sets are from the voice search domain.

4.1. Finding good ensembles

To investigate which types of acoustic models combine well, we evaluated various ensembles on French (fr-FR) and British

English (en-GB) recognition tasks. Specifically, we considered two-model combinations from a pool that varied architecture and training method, i.e. {LSTM, CLDNN} architecture with {CE, sMBR} training. As described in Section 2.2, an oracle can be used to estimate which model combinations will be the most complementary. Figure 2 shows the WERs of each ensemble with different combination weights, as well as the WER of its oracle combination (dotted lines). The x-axis shows the weight w_1 of the first model in the ensemble, the weight of the second model is $w_2 = 1.0 - w_1$. Thus, the most left and the most right points (0.0 and 1.0 on the x-axis) correspond to the WERs of the single models and points in-between to their weighted combinations.

As the results show, all model combinations yield better recognition accuracy than their constituent models individually, which shows that our simple late combination strategy is effective. In both French and British English tasks, the best model combinations are those that mix models with different architectures. The French results demonstrate that it can be advantageous to mix models of different training methods as well: The lowest WERs overall are achieved by combining LSTM-sMBR and CLDNN-CE models, which outperform the LSTM-sMBR and CLDNN-sMBR pair. As expected, it appears such diverse models make systematically different errors, allowing them to compensate for one another in combination.

In most cases, the order of oracle WERs of each combination corresponds to how well the real, weighted ensembles performed. This indicates that we can use the oracle WER to anticipate which model combinations will work well in practice. As discussed previously, such oracles can be computed easily for a given test set, needing only ASR evaluations of each model individually. As can be seen in both languages, the oracle performance is always better than the best performance of a weighted combination. This indicates that there is space for improvement of the combination function, e.g. by using a non-linear combination. Parameters of such a combination function can be learned during the training phase.

4.2. Knowledge distillation

Multi-model ensembles are typically too resource-intensive to deploy in real ASR systems. Thus, after finding ensembles that work best (using the methodology from the previous section), we used the procedure discussed in Section 3.1 to distill the ensembles to single student models. We evaluated our approach on ASR tasks in five languages: British English (en-GB), US Spanish (es-US), French (fr-FR), Korean (ko-KR) and Brazilian Portuguese (pt-BR). In all experiments, the student model was a CLDNN model as described in [13].

Table 1 shows WERs of best ensembles with their respective combination weights, student models after CE training, and the same students after additional sMBR training. We evaluated combination of up to four models (including LSTM-CE, LSTM-sMBR, CLDNN-CE and CLDNN-sMBR models) and chose the combinations with the highest performance, which are presented in the table. The table also shows the relative change of WER compared to a baseline, which is the best single candidate model available for the given language (normally a CLDNN-sMBR model). As we see, all of the ensembles lead to performance gains, with largest gains on fr-FR (8.0% improvement) and ko-KR (7.2% improvement).

As the results show, the student models yield significant improvements over the baseline single-CLDNN models. After CE distillation alone, the student models give 2.4-6.2% better WER than their respective baselines. Furthermore, in most languages, the student performance improved further after sMBR fine-tuning. In es-US and pt-BR, the final student models even surpassed the performance of their teacher ensembles, achieving 8.9% and 3.6% relative WER reduction, respectively. As mentioned previously, good model initialization was important for achieving low WERs. For example, the final en-GB student model had 13.5 WER when initialized with random weights (vs. 12.6 when initialized from an existing model).

Altogether, the full recipe produces single CLDNN models that substantially outperform baselines that were trained on hard labels. This is particularly impressive given that the student and baseline models have the same architecture and identical numbers of parameters.

Ensemble	**Baseline**	**Ensemble**	**Student WER**	
(bold: initializer)	**WER**	**WER**	**CE**	**CE+sMBR**
en-GB 0.5 · LSTM (sMBR) **0.4 · CLDNN (sMBR)** 0.1 · CLDNN (CE)	13.1	12.6 (-3.8%)	12.6 (-3.8%)	12.6 (-3.8%)
es-US 0.5 · LSTM (sMBR) **0.5 · CLDNN (CE)**	11.8	11.2 (-5.2%)	11.3 (-3.9%)	10.7 (-8.9%)
fr-FR 0.5 · LSTM (sMBR) **0.5 · CLDNN (CE)**	16.2	14.9 (-8.0%)	15.2 (-6.2%)	15.0 (-7.4%)
ko-KR 0.4 · LSTM (sMBR) **0.3 · CLDNN (sMBR)** 0.3 · CLDNN (CE)	11.1	10.3 (-7.2%)	10.6 (-4.5%)	10.5 (-5.5%)
pt-BR 0.7 · LSTM (sMBR) **0.3 · CLDNN (CE)**	11.7	11.3 (-3.2%)	11.4 (-2.4%)	11.2 (-3.6%)

Table 1: Knowledge distillation results. WERs of teacher ensembles and CLDNN student models after CE distillation and after additional sMBR training. Baselines are the single-best models from each ensemble. CE distillation alone always improves over the baseline, and sMBR usually gives additional gains.

5. Conclusion

This work found a simple but powerful class of acoustic model ensembles consisting of LSTM and CLDNN models built with different training objectives. We developed an oracle-based measure to identify the best-performing ensembles and used knowledge distillation to train single CLDNN models having close to – or exceeding – the predictive power of those ensembles. In contrast to some prior work, our method is easy to implement on top of standard training tools and produces single models that are small and efficient enough to deploy in real ASR systems. Our recipe gives up to 8.9% relative WER improvements over identically-structured baseline models, showing that it can train better models without the need to change the model architecture.

6. References

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

[2] Y. Zhao, J. Xue, and X. Chen, "Ensemble learning approaches in speech recognition," in *Speech and Audio Processing for Coding, Enhancement and Recognition*, T. Ogunfunmi, R. Togneri, and M. Narasimha, Eds. Springer New York, 2015, pp. 113–152.

[3] L. Deng and J. C. Platt, "Ensemble deep learning for speech recognition." in *INTERSPEECH*. ISCA, 2014, pp. 1915–1919.

[4] H. Soltau, G. Saon, and T. N. Sainath, "Joint training of convolutional and non-convolutional neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 5572–5576.

[5] T. N. Sainath, B. Kingsbury, A. Mohamed, G. E. Dahl, G. Saon, H. Soltau, T. Beran, A. Y. Aravkin, and B. Ramabhadran, "Improvements to deep convolutional neural networks for lvcsr," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 315–320.

[6] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression." in *Knowledge Discovery and Data Mining (KDD), 2006 ACM SIGKDD Conference On*, T. Eliassi-Rad, L. Ungar, M. Craven, and D. Gunopulos, Eds. ACM, 2006, pp. 535–541.

[7] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network." *Computing Research Repository (CoRR)*, vol. abs/1503.02531, 2015.

[8] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 7893–7897.

[9] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *Computing Research Repository (CoRR)*, vol. abs/1412.6550, 2014.

[10] J. Li, R. Zhao, J.-T. Huang, and Y. Gong, "Learning small-size dnn with output-distribution-based criteria." in *INTERSPEECH*. ISCA, 2014, pp. 1910–1914.

[11] W. Chan, N. R. Ke, and I. Lane, "Transferring knowledge from a rnn to a dnn." *Computing Research Repository (CoRR)*, vol. abs/1504.01483, 2015.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, 1997.

[13] T. N. Sainath, O. Vinyals, A. W. Senior, and H. Sak, "Convolutional, long short-term memory, fully connected deep neural networks." in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4580–4584.

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

[15] L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy," *Machine Learning*, vol. 51, no. 2, pp. 181–207, 2003.

[16] R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "Ensemble diversity measures and their application to thinning." *Information Fusion*, vol. 6, no. 1, pp. 49–62, 2005.

[17] H. Schwenk, "Using boosting to improve a hybrid hmm/neural network speech recognizer." in *Acoustics, Speech and Signal Processing (ICASSP), 1999 IEEE International Conference on*. IEEE, 1999, pp. 1009–1012.

[18] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[19] H. Sak, O. Vinyals, G. Heigold, A. W. Senior, E. McDermott, R. Monga, and M. Z. Mao, "Sequence discriminative distributed training of long short-term memory recurrent neural networks." in *INTERSPEECH*. ISCA, 2014, pp. 1209–1213.

[20] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, A. Senior, P. Tucker, K. Yang, Q. V. Le *et al.*, "Large scale distributed deep networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1223–1231.