INTERSPEECH 2020
October 25–29, 2020, Shanghai, China

Continual Learning for Multi-Dialect Acoustic Models

Brady Houston, Katrin Kirchhoff

Amazon, Seattle, Washington, USA
{hstbrady, katrinki}@amazon.com

Abstract

Using data from multiple dialects has shown promise in improving neural network acoustic models. While such training can improve the performance of an acoustic model on a single dialect, it can also produce a model capable of good performance on multiple dialects. However, training an acoustic model on pooled data from multiple dialects takes a significant amount of time and computing resources, and it needs to be retrained every time a new dialect is added to the model. In contrast, sequential transfer learning (fine-tuning) does not require retraining using all data, but may result in catastrophic forgetting of previously-seen dialects. Using data from four english dialects, we demonstrate that by using loss functions that mitigate catastrophic forgetting, sequential transfer learning can be used to train multi-dialect acoustic models that narrow the WER gap between the best (combined training) and worst (fine-tuning) case by up to 65%. Continual learning shows great promise in minimizing training time while approaching the performance of models that require much more training time.

Index Terms: speech recognition, acoustic modeling, multi-dialect

1. Introduction

Typically, acoustic models (AM) for speech recognition are trained on data from a single dialect, and correspondingly, used for recognition only on that same dialect. However, much work has been done to show that neural network acoustic models can be trained on data from multiple dialects. These approaches range from simple fine-tuning [1][2][3] to more complicated multi-task networks [4][5][3][6][7] and typically show that performance is at least as good as that of models trained with single-dialect data, if not better. While improved performance for a given dialect during recognition is desirable in and of itself, a single model which can recognize multiple dialects is also useful, e.g., a model that performs well across all English dialects spoken in the United Kingdom.

Despite the benefits of multi-dialect AMs, training an AM to recognize speech from multiple dialects can become costly as the number of dialects and the amount of training data per dialect increase. Furthermore, to add a new dialect to the model's recognition capabilities, the model cannot be simply fine-tuned on the new data; the entire model must be trained using data from all dialects, otherwise the model will undergo "catastrophic forgetting" and performance on previously-trained dialects will suffer [8]. Thus, the time to train the model (and thus the cost to business) increases linearly with each new dialect (considering all dialects to have a comparable amount of training data and compared to constant time for fine-tuning).

Fortunately, there are myriad approaches in the field of continual learning (CL) that have been shown to mitigate this catastrophic forgetting effect and provide a model that has good performance across several different, sequentially trained tasks (in this case, dialects). Two of the major thrusts in this area are regularization terms that are added to the loss function during training to prevent forgetting [8][7][9] and novel architectures [10][11][12]. These approaches have the benefit of not requiring that all data be present in order to train the model, i.e. new dialects can be added without retraining using data from all the previous dialects. Many of these approaches have been used in the fields of reinforcement learning and computer vision but have not yet been thoroughly applied to problems in speech recognition. At least one study has examined a few CL approaches to creating sequentially trained multi-dialect acoustic models with promising results, however, only a single transfer step from one dialect to three others was explored [13].

In this work, several approaches are investigated to apply CL to the task of creating a sequentially-trained neural network AM that can be used for recognition in a hybrid BLSTM-HMM ASR system across multiple English dialects. Instead of only creating models that can recognize two dialects, we train a single acoustic model that can recognize up to four different dialects with performance that covers up to 65 % of the difference between a fine-tuned model (worst case) and a model trained using all data (best case). These results highlight the utility of continual learning approaches in ASR, particularly as more and more data becomes available for training acoustic models.

2. Methods

2.1. Multidialect acoustic models

Many different approaches exist for creating multi-dialect acoustic models (and even pre-date neural acoustic models, see [14] for example), usually to achieve one or both of the following goals: 1) an ASR system capable of predicting text from speech in multiple dialects using a single acoustic model, or 2) leveraging data from other dialects to improve an acoustic model that is intended for use with a single dialect. Though many of these methods are quite complex, involving multi-objective training and/or complex features, two of the most common approaches are combined training and transfer learning. These methods are simple, yet effective, and will be used in this work as a comparison for CL approaches.

Combined training entails training a single model with all layers being trained on data from all dialects. This type of training requires that all dialects share a senone set. This can be achieved by explicitly using the same phoneset and cart tree for clustering (approach used in this work) [15], mapping the phonesets from each dialect to a "canonical" phoneset using expert knowledge [16][3], or using more complicated, possibly data-driven methods [17][18]. Combined training has been shown to improve performance across the dialects included in the model, compared to training on single dialects, particularly when one or more dialects have a scarcity of data [3]. A variant of combined training involves using a multi-task approach, where each dialect has it's own output layer, but intermedi-

http://dx.doi.org/10.21437/Interspeech.2020-1797

ate layers are shared (shared hidden layers) [4]. In this work, we use simple combined training instead of an SHL approach because preliminary results showed them to similar in performance.

In dialectal transfer learning, a model that has previously been trained on data from one dialect (typically one that has an abundance of data, if possible) is used to seed the initialization weights for a new model, which is then trained on data from another dialect [1][2][3]. If the dialects share the same senone and feature sets, all layers can be "fine-tuned". Otherwise, hidden layers can be reused, while input and/or output layers can be re-initialized to random weights and trained from scratch on the new data. This approach often yields improved performance on the fine-tuned dialect, however, performance on the dialect(s) used to train the seed model degrades, usually significantly. This catastrophic forgetting effect prevents the model from being useful as a true, multi-dialect model.

2.2. Continual learning

The goal of CL is to sequentially expose a model to various tasks in such a way that can maintain good performance on all the tasks seen by the model. In order to do this, the catastrophic forgetting effect must be mitigated so that performance on previous tasks is not sacrificed for performance on the current task. There are many approaches that can achieve this effect, but one of the most common is to include a forgetting penalty in the loss function (in the case of this work, standard frame-wise cross-entropy) during training. The two CL penalties investigated in this work are elastic weight consolidation (EWC) [8] and learning without forgetting (LWF) [7]. Both of these methods incorporate model information from previous steps during training at the current step to prevent loss of performance on those steps. These approaches will be discussed in more detail below.

2.2.1. Elastic weight consolidation

During neural network training on a task (task 1), model parameters are incrementally updated to minimize the specified loss function. This moves the model parameters to a point in an area of the parameter space that is yields good performance for task 1. If the model is then trained on a second task (task 2), the model parameters freely move to a point in the parameter subspace that yields good performance for task 2. There is no guarantee that this second point gives good performance on task 1. However, it is possible that the good-parameter subspaces for task 1 and task 2 share some overlap. By constraining the parameters to remain in the "good" subspace for task 1 during task 2 training, the model can arrive at a point where performance on both tasks is good. This constraint can be achieved by including a quadratic (elastic) penalty term that minimizes the distance between the new model parameter values for task 2 and the old values for task 1:

$$J_{elastic} = \lambda \sum_i (\theta_{i,t_1} - \theta^*_{i,t_2})^2 \quad (1)$$

where i indexes the model parameters, t_1 and t_2 are the previous and current task, respectively, $*$ superscript indicates what is trainable, and λ balances remembering of the previous task with learning the current task. The notation for equation 1 unrolls all of the weight matrices in a neural network into a single, Nx1 vector. However, this penalty term does not account for the fact that not all model parameters are equally important to model performance on task 1, and thus not all parameters need to be "remembered".

Information about which model parameters are most important to task 1 are incorporated into the model's posterior distribution for the data from task 1. This distribution can be approximated as Gaussian with a mean θ_i and a standard deviation using the diagonal of the Fisher information matrix. The Fisher information matrix is the second derivative of the model's loss, and thus reflects the "sharpness" of the model's parameters, as estimated by the data. Including the Fisher diagonal values in the quadratic penalty term from equation 1 allows the model to only remember the parameters that are most important for performance on task 1, and yields the EWC loss term:

$$J_{ewc} = \lambda \sum_i F_i (\theta_{i,t_1} - \theta^*_{i,t_2})^2 \quad (2)$$

where F_i are the Fisher diagonal values for task t_1. Only remembering the most critical parameters for the previous task increases the model's capacity to modify other parameters to achieve good performance on task t_2. The Fisher diagonal value for a given parameter can be approximated itself by summing the squared gradients for that parameter across several (many) training examples.

This EWC loss can be extended to more than two tasks in multiple ways. The most straightforward way is to simply constrain a new task to each of the old tasks by keeping individual Fisher diagonals and model parameters from each step:

$$J_{ewc} = \sum_t \lambda \sum_i F_{i,t} (\theta_{i,t} - \theta^*_i)^2 \quad (3)$$

where t denotes the previous tasks. However, this requires maintaining Fisher diagonal and model parameters across all tasks. An alternative method, known as "online EWC", is to "re-center" the Gaussian approximation by only using the model parameters from the most recent task and keep a running Fisher estimation across all tasks (as opposed to using model parameters and Fisher values from each previous task) [12]. Online EWC is the approach used in this work.

2.2.2. Learning without forgetting

As discussed above, EWC explicitly seeks to constrain model parameters to a subspace where performance is optimal on both previous and current tasks. LWF, on the other hand, tries to directly maintain the relationship between input values and output labels from previous tasks. It does so by feeding training data from the current task (task 2) into the model from a previous task (task 1), and constraining the output of the model from task 2 to be "close" to the output of the model from task 1. It is thus an extension of model student-teacher knowledge distillation [19] that has already seen some use in ASR [20]. This distillation is achieved by using cross-entropy with the outputs from the model from task 1 as soft targets:

$$J_{lwf} = -\lambda \sum_i p_{t_1,i} \log(x_{t_2,i}) \quad (4)$$

where $x_{t_2,i}$ are the posterior probabilities from the model for task 2, $p_{t_1,i}$ are the posterior probabilities from the model for task 1, and λ is a hyperparameter that balances remembering the previous task with learning the new task (typically, the standard frame-wise cross entropy is scaled by $1 - \lambda$). By adding this penalty term to the loss for training the model for task 2, the model is able to maintain the input-output relationship learned

Table 1: *Sequential training using continual learning techniques ("CL") covers up to 65.2% of the WER gap between fine-tuning ("FT"; worst case) and combined training ("Comb"; best case) across three different transfer steps (en-US → en-GB, en-GB → en-AU, en-AU → en-IN). Several different hyperparameters and continual learning methods were explored, and these results show the best-performing set as determined by the grand average WER across all three steps. The seed model (en-US) for all experimental conditions had a WER of 13.7%. See eqn. 7 for explanation of WER "gap coverage".*

	en-US	en-GB	Avg	en-US	en-GB	en-AU	Avg	en-US	en-GB	en-AU	en-IN	Avg
FT	21.1	16.0	18.6	19.7	24.9	26.9	23.8	35.0	50.6	47.9	24.3	39.5
Comb	13.7	15.8	14.7	13.5	15.6	26.2	18.4	13.5	15.5	26.0	24.9	20.0
CL	15.1	17.6	16.3	16.0	19.7	28.0	21.2	18.2	26.0	32.9	29.0	26.7
Covered			58.1%				47.6%					65.2%

by the model for task 1. "Softer" targets for task 1 can be obtained by applying the softmax temperature T to the logits z:

$$\frac{\exp(z_j/T)}{\sum_i \exp(z_i/T)} \tag{5}$$

allowing the model for task 2 to learn more of the target distribution. We have found that increasing this temperature can lead to much larger search spaces and increased latency during decoding, so in this work we use $T = 1$.

Like EWC, this LWF penalty can be extended to multiple tasks in several ways. Again, the most straightforward is to use soft targets for each of the previously seen tasks:

$$J_{lwf} = \sum_t -\lambda \sum_i p_{t,i} \log(x_i) \tag{6}$$

where $p_{t,i}$ denotes the previous tasks. However, this requires keeping an increasing number of target distributions (or feeding training examples through an increasing number of previous models during training) as the number of tasks increases. An alternative is to just use the model/targets from the most recent task, instead of all previous tasks, as this model should already contain information about the input-output relationships from all previous tasks. The danger with this approach is that the model will be biased to more recently-visited tasks, and so this risk must be balanced with the training time/complexity that arise from maintaining targets from all tasks. In this work, we opt for the second approach and only use target outputs from the most recent previous task.

3. Data and Experimental Setup

For this work, conversational telephony speech and audio data from four different English dialects were used: United States (US), Great Britain (GB), Australia (AU) and India (IN). For acoustic model training, 1000 hours of en-US data were used; for all other dialects, 300 hours of training data were used. All dialects had 15 hours of data set aside for validation and 15 hours for testing. A previously-trained en-US model was used to obtain forced alignments for all dialects. Previously-trained, dialect-specific 4-gram language models with KN-smoothing were used for decoding.

All experiments (including combined, fine-tuning and CL methods) used the following protocol for neural network training. First, 40-dim MFCC features were extracted from audio files at 10 ms intervals using a 25 ms window. Features were then normalized to zero-mean and unit variance using per-segment statistics. Acoustic models consisted of BLSTM with a fully-connected layer as the output to predict senone posterior probabilities (all dialects used the same number of senones). The number of hidden layers was varied between 4 and 6 to examine its effect on the ability of the network to retain knowledge about previous tasks. Standard framewise cross-entropy was used as the loss function during training (in addition to the continual-learning loss functions described above). For EWC, the λ hyperparameter was chosen from values of 100, 500, and 1000. For LWF, the λ value was chosen from values of 0.4, 0.5 and 0.6. Dropout was used to prevent overfitting, and values were chosen from 0.1, 0.2 and 0.3 to examine their effect on performance. Learning rate warmup was used during training, and ranged from 0.0001 to 0.001 over 10 epochs. Learning rate was decayed based on training error, with a decay factor of 0.75 and a threshold of 0. ADAM with Nesterov momentum was used for model optimization, and the minibatch size was 30000 frames. Training was performed using eight GPUs in a single machine. Model convergence was determined by five consecutive steps with an absolute difference in frame error rate less than 0.00005, and typically required 50-80 epochs to converge. Sequence discriminative training was not used after CE training, but may be explored in future work. After convergence, model checkpoints from every 10 epochs and the final epoch were used to calculate WER on test data from all previously-trained dialects to find the best seed model for the next round of sequential training. This approach was because simply picking the epoch with the lowest frame error rate would bias the model to the most recently visited training step.

To examine the efficacy of different continual learning methods, the baseline en-US acoustic model was sequentially re-trained on the three remaining dialects, in the order en-US → en-GB → en-AU → en-IN. At each new step, the best-performing acoustic model (in terms of average WER on test data from all dialects used thus far in training) from the previous step was used to seed the weights for the new model.

4. Results and Discussion

This work examined continual learning techniques for avoiding catastrophic forgetting during sequential training of a neural network acoustic model across several different dialects. For comparison, best case (trained on all dialects at a given step) and worst case (fine-tuned only on dialect at a given step) models were also trained for each step. As expected, sequential fine-tuning of the AM resulted in catastrophic forgetting at each step (Table 1), as seen by an increase in WER of previously-seen dialects: at the final step (en-AU → en-IN), performance on the three previous dialects(en-US, en-GB and en-AU) was, on average, about 150% (range: 78.4%-215.7%) worse than at each dialect's respective fine-tuning step and 71.2% worse than when all four dialects are trained together.

In contrast, when CL terms were added to the training loss function to prevent this catastrophic forgetting, performance was much better than fine-tuning. To measure this effect, we

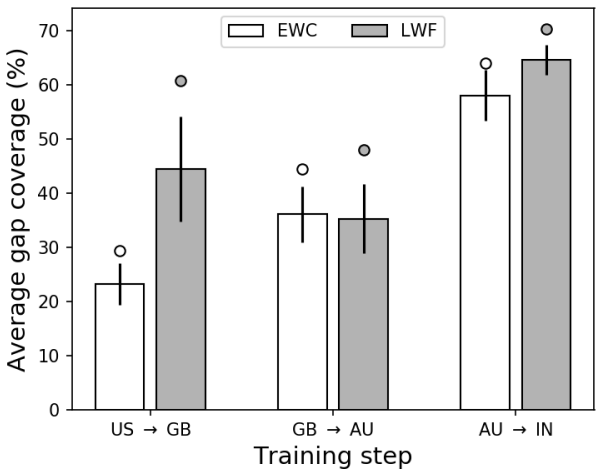

Figure 1: *Distributions of model performance (in terms of gap-coverage) using LWF and EWC to prevent catastrophic forgetting. Each bar summarizes all experimental runs for a given CL method (white = EWC, gray = LWF). These runs consist of different combinations of CL weight, dropout value, number of LSTM layers, and transfer step. Each bar depicts the average gap-coverage and standard deviation. The best performing CL method at each step (shown by filled circle above bar) is LWF.*

introduce the gap-coverage metric:

$$\text{GAP COVERED} = 1 - \frac{\text{WER}_{CL} - \text{WER}_{Comb}}{\text{WER}_{FT} - \text{WER}_{Comb}} \quad (7)$$

which is a measure of how close the CL method (WER_{CL}) is to attaining the performance of combined training. For each CL experiment (combination of transfer step, CL method, CL weight, dropout, number of layers), there is an equivalent combined experiment (same dialects, dropout, number of layers) which represents the best case performance (WER_{Comb} in equation 7) and an equivalent fine-tuning experiment (same dialect, dropout, number of layers) which represents the worst case performance (WER_{FT} in equation 7. All WERs are averaged across the dialects at a given step, and the ideal CL method would be able to achieve a gap-coverage of 100% (or greater). As an example, suppose at the training step en-GB $\rightarrow$ en-AU, combined training gives a model with an average WER of 25% across en-US, en-GB and en-AU, fine-tuning from en-GB $\rightarrow$ en-AU gives a model with an average WER of 35% across the three dialects, and using a CL method gives a model with an average WER of 28% across the three dialects. This would yield a gap-coverage value of 70%.

At best, using CL methods during training resulted in gap coverage of up to 65.2% at the final step (en-AU $\rightarrow$ en-IN). The gap covered in the first two steps (en-US $\rightarrow$ en-GB: 58.1%; en-GB $\rightarrow$ en-AU: 47.6%) was not quite as large as that covered in the final step. There are several likely contributions to this observation, but it is largely due to the much greater catastrophic forgetting observed at the final step (see Table 1, "FT"). This larger catastrophic forgetting effect at the final step is in turn likely attributed to the en-IN dialect having a much different sound quality than the other three dialects. This effect could also be related to the fact that en-IN is the final step, and thus en-US and en-GB are being remembered less since the fine-tuning has gone through several iterations (despite using CL methods). Future work might visit the dialects in different orders in order to tease out the causes of this phenomenon.

As part of this work, several hyperparameters were tuned; to explore the interplay of model capacity and the effect of CL methods, dropout and the number of LSTM layers were varied. Additionally, the weight given to the CL term in the loss function was also varied to see what effect it has on the performance on previous tasks vs the most recently visited task. For a fair comparison between CL methods and to prevent exploding parameters spaces, we selected three values for each of these hyperparameters, for a grid-search space size of nine parameter combinations for each CL method. The ranges for these penalty terms were chosen based on preliminary experiments not discussed in this work.

At each step, using LWF as the CL penalty term gave the best results (in terms of gap covered between combined training and fine-tuning) (Figure 1; see filled circle above each bar for best result at each step). One possible reason for this is that the relatively small range of values for the CL weights was not optimal for EWC, but this seems unlikely to have played a large role due to the relatively tight clustering of performance from models using EWC. Another explanation is that LWF's ability to directly maintain the input-output relationship for previously seen tasks is simply more amenable to speech than EWC's attempt to indirectly maintain this relationship by constraining model weights. This might be explained by the fact that ASR AMs typically have a very high number of classes at the output layer, so restricting the model at a less granular level is more effective than constraining model parameters.

Despite LWF providing better results at each step, the actual gap covered by both LWF and EWC methods showed large variations. The dropout and number of LSTM layers didn't appear to have much of an effect on performance when using EWC or LWF at any of the steps, but generally more layers and smaller dropout seemed to be slightly better. This may explained by the fact that the best and worse case comparison models had the same dropout and LSTM layers, and so the effect of these parameters was consistent whether CL was employed or not.

The effect of varying the CL weight for LWF and EWC was more complex, and there was not a consistent effect seen across steps. At the first step, the variation due to the LWF weight was very large, with the best results coming from using a smaller teacher weight. This observation was not seen when using EWC, or while using either of the CL methods at any of the other steps. As discussed above, there is likely a complex interplay between the similarity of the dialects already visited and the new dialect, as well as the CL method and weight. More research, such as varying the CL weight value at each step and the order of steps is necessary to tease out the effects of each.

5. Conclusions

In this work, we explore two different continual learning approaches, namely elastic weight consolidation and learning without forgetting, to sequentially training a multi-dialect, neural network acoustic model. Using these methods, we are able to sequentially train a single acoustic model with performance across four different English dialects that covers up to 65% of the gap between sequential fine-tuning and training on all data. These results show the utility of applying continual learning methods to ASR, particularly as data sets become larger and larger. Future work will explore extensions of these methods and applications to other continual learning tasks in ASR beyond multi-dialect modeling.

6. References

[1] N. T. Vu, W. Breiter, F. Metze, and T. Schultz, "An investigation on initialization schemes for multilayer perceptron training using multilingual data and their effect on asr performance," 2012.

[2] A. Ghoshal, P. Swietojanski, and S. Renals, "Multilingual training of deep neural networks," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 7319–7323.

[3] H. Arsikere, A. Sapru, S. Garimella, and A. M. Learning, "Multi-dialect acoustic modeling using phone mapping and online i-vectors," *Interspeech*, pp. 2125–2129, 2019.

[4] J. T. Huang, J. Li, D. Yu, L. Deng, and Y. Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 7304–7308, 2013.

[5] G. Heigold, V. Vanhoucke, A. Senior, P. Nguyen, M. Ranzato, M. Devin, and J. Dean, "Multilingual acoustic models using distributed deep neural networks," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, pp. 8619–8623, 2013.

[6] T. Viglino, P. Motlicek, and M. Cernak, "End-to-end Accented Speech Recognition," *Interspeech*, pp. 1–5, 2019.

[7] Z. Li and D. Hoiem, "Learning without Forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 12, pp. 2935–2947, 2018.

[8] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks." *Proceedings of the National Academy of Sciences of the United States of America*, no. 13, pp. 3521–3526.

[9] S. Sodhani, S. Chandar, and Y. Bengio, "Towards Training Recurrent Neural Networks for Lifelong Learning," pp. 1–35, 2018. [Online]. Available: http://arxiv.org/abs/1811.07017

[10] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive Neural Networks," 2016. [Online]. Available: http://arxiv.org/abs/1606.04671

[11] X. Li, Y. Zhou, T. Wu, R. Socher, and C. Xiong, "Learn to Grow: A Continual Structure Learning Framework for Overcoming Catastrophic Forgetting," 2019. [Online]. Available: http://arxiv.org/abs/1904.00310

[12] J. Schwarz, J. Luketina, W. M. Czarnecki, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, and R. Hadsell, "Progress & compress: A scalable framework for continual learning," *35th International Conference on Machine Learning, ICML 2018*, vol. 10, pp. 7199–7208, 2018.

[13] S. Ghorbani, S. Khorram, and J. H. L. Hansen, "Domain Expansion in DNN-based Acoustic Models for Robust Speech Recognition," 2019. [Online]. Available: http://arxiv.org/abs/1910.00565

[14] N. Beringer, F. Schiel, and P. Regel-Brietzmann, "German regional variants-a problem for automatic speech recognition?" *Fifth International Conference on Spoken Language Processing*, 1998.

[15] M. Mueller and A. Waibel, "Using Language Adaptive Deep Neural Networks for Improved Multilingual Speech Recognition," *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, pp. 167–172, 2015. [Online]. Available: http://www.mt-archive.info/15/IWSLT-2015-muller-2.pdf

[16] T. Schultz and A. Waibel, "Multilingual and crosslingual speech recognition," *DARPA Workshop on Broadcast News ...*, pp. 3–6, 1998.

[17] X. Li, S. Dalmia, J. Li, M. Lee, P. Littell, J. Yao, A. Anastasopoulos, D. R. Mortensen, G. Neubig, A. W. Black, and F. Metze, "Universal Phone Recognition with a Multilingual Allophone System," 2020. [Online]. Available: http://arxiv.org/abs/2002.11800

[18] X. V. P. M. A. W. Mohamed Elfeky, Meysam Bastani, "TOWARDS ACOUSTIC MODEL UNIFICATION ACROSS DIALECTS," *IEEE SLT*, pp. 624–628, 2016.

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," pp. 1–9, 2015. [Online]. Available: http://arxiv.org/abs/1503.02531

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