MULTI-DIALECT SPEECH RECOGNITION WITH A SINGLE SEQUENCE-TO-SEQUENCE MODEL

Bo Li, Tara N. Sainath, Khe Chai Sim, Michiel Bacchiani,
Eugene Weinstein, Patrick Nguyen, Zhifeng Chen, Yonghui Wu, Kanishka Rao

Google Inc., USA

{boboli,tsainath,khechai,michiel,weinstein,drpng,zhifengc,yonghui,kanishkarao}@google.com

ABSTRACT

Sequence-to-sequence models provide a simple and elegant solution for building speech recognition systems by folding separate components of a typical system, namely acoustic (AM), pronunciation (PM) and language (LM) models into a single neural network. In this work, we look at one such sequence-to-sequence model, namely listen, attend and spell (LAS) [1], and explore the possibility of training a single model to serve different English dialects, which simplifies the process of training multi-dialect systems without the need for separate AM, PM and LMs for each dialect. We show that simply pooling the data from all dialects into one LAS model falls behind the performance of a model fine-tuned on each dialect. We then look at incorporating dialect-specific information into the model, both by modifying the training targets by inserting the dialect symbol at the end of the original grapheme sequence and also feeding a 1-hot representation of the dialect information into all layers of the model. Experimental results on seven English dialects show that our proposed system is effective in modeling dialect variations within a single LAS model, outperforming a LAS model trained individually on each of the seven dialects by 3.1~16.5% relative.

Index Terms— multi-dialect, sequence to sequence, adaptation

1. INTRODUCTION

Dialects are defined as variations of the same language, specific to geographical regions or social groups. Although they share many similarities, there are usually large differences at several linguistic levels; amongst others: phonological, grammatical, orthographic (e.g., "color" *vs.* "colour") and very often different vocabularies. As a result, automatic speech recognition (ASR) systems trained or tuned for one specific dialect perform poorly when tested on another dialect of the same language [2]. In addition, systems simultaneously trained on many dialects fail to generalize well for each individual dialect [3]. Inevitably, multi-dialect languages pose a challenge to ASR systems. If enough data exists for each dialect, a common practice is to treat each dialect independently [2, 4, 5]. Alternatively, in cases where dialects are resource-scarce, these models are boosted with data from other dialects [3, 6]. In the past, there have been many attempts to build multi-dialect/language systems. The usual approach has been to define a common set of universal phone models [7, 8, 9] with appropriate parameter sharing [6] and train it on data from many languages with eventual adaptation on the data from the language of interest [10, 11, 12, 13]. [11, 14] developed similar neural network models with language independent feature extraction and language dependent phonetic classifiers. [12] further investigated a grapheme-based multi-dialect model with dialect-dependent phoneme recognition as secondary tasks. Adaptation techniques such as MLLR and MAP are commonly used for Gaussian mixture model based systems [13]; but for neural network based models, adaptation by continued training on dialect-specific data works well [12].

We believe it has been challenging to build a universal multi-dialect model for conventional ASR systems because these models still require a separate pronunciation and language model (PM and LM) per dialect, which are trained independently from the multi-dialect acoustic model (AM). Therefore, if the AM predicts an incorrect set of sub-word units from the wrong dialect, errors get propagated to the PM and LM.

Sequence-to-sequence models provide a simple and elegant solution to the ASR problem by learning and optimizing a single neural network for the AM, PM and LM [15, 16, 17, 18, 19, 20]. We believe such a model is very attractive to look at for building a single multi-dialect system specifically for this reason. Training a multi-dialect sequence-to-sequence model is simple, as it requires simply pooling all the grapheme symbols together. In addition, the AM, PM and LM variations are jointly modeled across dialects [21]. The simplicity and joint optimization make it a good candidate for training multi-dialect systems.

In this work, we adopt the attention-based sequence-to-sequence model, namely listen, attend and spell (LAS) [1], for multi-dialect modeling. It has shown good performance compared to other sequence-to-sequence models for single dialect tasks [20]. Our goal here is to build a single LAS model for seven English dialects. We start by simply pooling all the data together. For English the grapheme set is shared across dialects, nothing needs to be modified for the output. Although this model gives acceptable performance on each dialect, it falls behind the ones independently fine-tuned on each dialect.

In the literature, adaptation methods [22, 23] are typically used to integrate dialect-specific information into the system. We hypothesize that by explicitly providing dialect information to the LAS model we should be able to bridge the gap between the dialect-independent and dialect-dependent models. Firstly, we use dialect information in the output by introducing an artificial token into the grapheme sequence [24]. The LAS model needs to learn both grapheme prediction and dialect classification. Secondly, we feed the dialect information as input vectors to the system. It can be either used as an extra information vector appended to the inputs of each layer or as weight coefficients for cluster adaptive training (CAT) [25]. Our experimental results show that using dialect information can elevate the performance of multi-dialect LAS system to outperform dialect-dependent ones. The proposed system has several attractive benefits: 1) simplicity: no changes are required for the

model and scaling to more dialects is trivial - by simply adding more data; 2) improvement for low-resource dialects: in the multi-dialect system, the majority of the parameters are implicitly shared by all the dialects, which forces the model to generalize across dialects during training. It is observed that the recognition quality on the low resource dialect is significantly improved.

2. MULTI-DIALECT LAS MODEL

The LAS [1] model consists of an *encoder* (akin to an acoustic model), a *decoder* (like a language model) and an *attention* model which learns an alignment between the encoder and decoder outputs. The encoder is normally a stack of recurrent layers; in this study we use 5 layers of unidirectional long short term memory (LSTM) [26]. The decoder is a neural language model, consisting of 2 LSTM layers. The attention module takes in the decoder's lowest layer's state vector from the previous time step and estimates attention weights for each frame in the encoder output in order to compute a single context vector. The context vector is then input into the decoder network, along with the previously predicted label from the decoder to generate logits from the final layer in the decoder. Finally, these logits are input into a softmax layer, which outputs a probability distribution over the label inventory (i.e., graphemes), conditioned on all previous predictions. In conventional LAS models, the label inventory is augmented with two special symbols, <sos>, which is input to the decoder at the first time-step, and <eos>, which indicates the end of a sentence. During inference, the label prediction process terminates when the <eos> label is generated.

The baseline multi-dialect LAS system is built by simply pooling all the data together. The output targets consist of 75 graphemes for English, which are shared across dialects. In the following section, we describe how we can improve the baseline multi-dialect LAS model by providing dialect information. We assume this information is known in advance or can be easily obtained [5]. We believe explicitly providing such dialect information would be helpful to improve the performance of the multi-dialect LAS model. We discuss three ways of passing the dialect information into the LAS model, namely feeding it as output targets, as input vectors, or directly factoring the encoder layers based on the dialect.

2.1. Dialect Information as Output Targets

A common way to make the LAS model aware of the dialect is through multi-task learning [27]. We can add an extra dialect classification cost to the training and regularize the model to learn hidden representations that are effective for both dialect classification and grapheme prediction. However, this requires having two separate objective functions that are weighted, and deciding the optimal weight for each task is a parameter that needs to be swept [27].

A simpler approach, similar to [24], is to expand the label inventory of our LAS model to include a list of special symbols, each corresponding to a dialect. For example, when including the British English, we add the symbol <en-gb> into the label inventory. In [24], the special symbol is added to the beginning of the target label sequence. For example, for a British accented speech utterance of "hello world", the conventional LAS model uses "<sos> h e l l o ␣ w o r l d <eos>" as the output targets; in the new setup the output target is "<sos> <en-gb> h e l l o ␣ w o r l d <eos>". The model needs to figure out which dialect the input speech is first before making any grapheme prediction.

In LAS, each label prediction is dependent on the history. Adding the dialect symbol at the beginning, we implicitly incur the dependency of the grapheme prediction on the dialect classification. When the model makes errors in dialect classification, it may hurt the grapheme recognition performance. In this study, we assume the correct dialect information is always available. Hence, we explore inserting the dialect symbol at the end of the label sequence as well. For the example utterance, the target sequence now become "<sos> h e l l o ␣ w o r l d <en-gb> <eos>". By inserting the dialect symbol at the end, the model still needs to learn a shared representation but avoids the unnecessary dependency and is less sensitive to dialect classification errors.

2.2. Dialect Information as Input Vectors

Another way of providing dialect information is to pass this information as an additional feature [28]. To convert the categorical dialect information into a real-valued feature vector, we investigate the use of 1-hot vectors, whose values are all '0' except for one '1' at the index corresponding to the given dialect, and data-driven embedding vectors whose values are learned during training. These dialect vectors can be appended to different layers in the LAS model. At each layer the dialect vectors are linearly transformed by the weight matrices and added to the original hidden activations before the nonlinearity. This effectively enables the model to learn dialect-dependent biases. We are mainly interested in two configurations: 1) adding it to the encoder layers, which effectively provides dialect information to help model the acoustic variations across dialects; and 2) appending it to the decoder layers, which models dialect-specific language model variations. Ultimately, we can also combine the two by feeding dialect vectors into both the encoder and the decoder.

2.3. Dialect Information as Cluster Coefficients

Another approach to modeling variations in the speech signal (for example dialects) which has been explored for conventional models is cluster adaptive training (CAT) [25]. We can treat each dialect as a separate cluster and use 1-hot dialect vectors to switch clusters; alternatively, we can use data-driven dialect embedding vectors as weights to combine clusters. A drawback of the CAT approach is that it adds extra network layers, which typically adds more parameters to the LAS model. In this study, our goal is to maintain simplicity of the LAS model and limit the increase in model parameters. Thus, we only tested a simple CAT setup for the encoder of the LAS model to compare with the input vector approaches discussed in the previous sections. Specifically, we use a few clusters to compensate activation offsets of the 4th LSTM layer based on the shared representation learned by the 1st LSTM layer, to account for the dialect differences. For each cluster, a single layer 128D LSTM is used with output projection to match the dimension of the 4th LSTM layer. The weighted sum of all the CAT bases using dialect vectors as interpolation weights is added back to the 4th LSTM layer's outputs, which are then fed to the last encoder layer.

3. EXPERIMENTAL DETAILS

Our experiments are conducted on about 40K hours of noisy training data consisting of 35M English utterances. The training utterances are anonymized and hand-transcribed, and are representative of Google's voice search traffic. It includes speech from 7 different dialects, namely America (US), India (IN), Britain (GB), South Africa (ZA), Australia (AU), Nigeria & Ghana (NG) and Kenya (KE). The amount of dialect-specific data can be found in Table 1. The training data is created by artificially corrupting clean utterances

using a room simulator, adding varying degrees of noise and reverberation such that the overall SNR is between 0 and 20dB [29]. The noise sources are from YouTube and daily life noisy environmental recordings. We report results on dialect-specific test sets, each contains roughly 10K anonymized, hand-transcribed utterances from Google's voice search traffic without overlapping with the training data. This amounts to roughly 11 hours of test data per dialect. All experiments use 80-dimensional log-mel features, computed with a 25ms window and shifted every 10ms. Similar to [30, 31], at the current frame, t, these features are stacked with 3 frames to the left and downsampled to a 30ms frame rate. In the baseline LAS model, the encoder network architecture consists of 5 unidirectional 1024D LSTM [26] layers. Additive attention [1] is used for all experiments. The decoder network is a 2-layer 1024D unidirectional LSTM. All networks are trained to predict graphemes, which have 75 symbols in total. The model has a total number of 60.6M parameters. All networks are trained with the cross-entropy criterion, using asynchronous stochastic gradient descent (ASGD) optimization [32], in TensorFlow [33]. The training terminates when the change of WERs on a dev set is less than a given threshold for certain number of steps.

Table 1: Number of utterances per dialect for training (M for million) and testing (K for thousand).

Dialect	**US**	**IN**	**GB**	**ZA**	**AU**	**NG**	**KE**
Train(M)	13.7	8.6	4.8	2.4	2.4	2.1	1.4
Test(K)	12.9	14.5	11.1	11.7	11.7	9.8	9.2

4. RESULTS

4.1. Pooling All Data

Firstly, we build a single grapheme LAS model on all the data together (`S1` in Table 2). For comparison, we also build a set of dialect-dependent models. Due to the large variations in the amount of data we have for each dialect, a lot of tuning is required to find the best model setup from scratch for each dialect. For the sake of simplicity, we take the joint model as the starting point and retraining the same architecture for each dialect independently (`S2` in Table 2). Instead of updating only the output layers [11, 14], we find reestimating all the parameters work better. To compensate for the extra training time the fine-tuning brings in, we also keep the baseline model training for similar extra number of steps; we do not find to improve the WER. Comparing the dialect-independent model (`S1`) with the dialect-dependent ones (`S2`), simply pooling the data together gives acceptable recognition performance, but having a language-specific model by fine-tuning still achieves better performance.

Table 2: WER (%) of dialect-independent (`S1`) and dialect-dependent (`S2`) LAS models.

Dialect	**US**	**IN**	**GB**	**ZA**	**AU**	**NG**	**KE**
`S1`	10.6	18.3	12.9	12.7	12.8	33.4	19.2
`S2`	**9.7**	**16.2**	**12.7**	**11.0**	**12.1**	33.4	**19.0**

4.2. Using Dialect-Specific Information

Our next set of experiments look at using dialect information to see if we can have a joint multi-dialect model improve performance over the dialect-specific models (`S2`) in Table 2.

4.2.1. Results using Dialect Information as Output Targets

We first add the dialect information into the target sequence. Two setups are explored, namely adding at the beginning (`S3`) and adding at the end (`S4`). The results are presented in Table 3. Inserting the dialect symbol at the end of the label sequence is much better than at the beginning, which eliminates the dependency of grapheme prediction on the erroneous dialect classification. `S4` is more preferable and outperforms the dialect-dependent model (`S2`) on all the dialects except for IN and ZA.

Table 3: WER (%) of inserting dialect information at the beginning (`S3`) or at the end (`S4`) of the grapheme sequence.

Dialect	**US**	**IN**	**GB**	**ZA**	**AU**	**NG**	**KE**
`S2`	9.7	**16.2**	12.7	**11.0**	12.1	33.4	19.0
`S3`	9.9	16.6	12.3	11.6	12.2	33.6	18.7
`S4`	**9.4**	16.5	**11.6**	**11.0**	**11.9**	**32.0**	**17.9**

4.2.2. Results using Dialect Information as Input Vectors

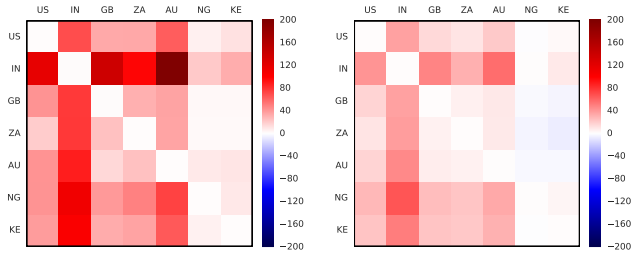
Table 4: WER (%) of feeding the dialect information into the LAS model's encoder (`S5`), decoder (`S6`) and both (`S7`). The dialect information is converted into an 8D vector using either 1-hot representation (`1hot`) or learned embedding (`emb`).

Dialect	**US**	**IN**	**GB**	**ZA**	**AU**	**NG**	**KE**
`S2`	9.7	16.2	12.7	11.0	12.1	33.4	19.0
`S5(1hot)`	9.6	16.4	11.8	10.6	10.7	31.6	18.1
`S5(emb)`	9.6	16.7	12.0	10.6	10.8	32.5	18.5
`S6(1hot)`	9.4	16.2	11.3	10.8	10.9	32.8	18.0
`S6(emb)`	9.4	16.2	**11.2**	10.6	11.1	32.9	18.0
`S7(1hot)`	**9.1**	**15.7**	11.5	**10.0**	**10.1**	**31.3**	**17.4**

Next we experiment with directly feeding the dialect information into different layers of the LAS model. The dialect information is converted into an 8D vector using either 1-hot representation or an embedding vector learned during training. This vector is then appended to both the inputs and hidden activations. We compare the usefulness of this dialect vector to the LAS encoder and decoder. From Table 4, feeding it to encoder (`S5`) gives gains on dialects with less data (namely GB, ZA, AU, NG and KE) and has comparable performance on US, but is still a bit worse on IN compared to the fine-tuned dialect-dependent models (`S2`). Similarly, we pass the dialect vector (using both 1-hot and learned embedding) to the decoder of LAS (`S6`). Table 4 shows that in this way the single multi-dialect LAS model outperforms the individually fine-tuned dialect-dependent models on all dialects except for IN, for which it obtains the same performance.

Comparing the use of 1-hot representation and learned embedding, we do not observe big differences for both the encoder and decoder. It is most likely the small dimensionality of the vectors used (*i.e.* 8D) that is insufficient to suggest any preference between the 1-hot representation and the learned embedding. In future, when scaling up to more dialects/languages, using embedding vectors instead of 1-hot to represent a larger set of dialects/languages could be a more economical way.

Feeding dialect vectors into different layers effectively enables the model to explicitly learn dialect-dependent biases. For the en-



(a) Encoder (`S5(1hot)`) (b) Decoder (`S6(1hot)`)

Fig. 1: Relative WER changes when feeding in wrong dialect vectors (rows) to the encoder or decoder for each test set (columns). The red and blue colors indicate the relative increase and decrease of WERs respectively and the white color means no change in WERs.

coder, these biases would help capture dialect-specific acoustic variations; while in the decoder, they can potentially address the language model variations. Experimental results suggest that these simple biases indeed help the multi-dialect LAS model. To understand their effects, we test system `S5(1hot)` and `S6(1hot)` with mismatched dialect vector on each test set. The relative WER changes are depicted in Figure 1. Each row represents the dialect vector fed into the model and each column corresponds to a dialect-specific test set. The white diagonal blocks are the "correct" setups, where we feed in the correct dialect vector on each test set. The red and blue colors represent the relative increase and decrease of WERs respectively. The darker the color is, the larger the change is. Comparing the effect on encoder and decoder, wrong dialect vectors degrades more on encoders, suggesting more acoustic variations across dialects than language model differences. Across different dialects, IN seems to have the most distinguishable characteristics. NG and KE, the two smallest dialects in this study, benefit more from the sharing of parameters as the performance varies little with different dialect vectors. This suggests the proposed model is capable of handling the unbalanced dialect data properly, learning strong dialect-dependent biases when there's enough data and sticking to the shared model otherwise. Another interesting observation is that, for these two dialects, feeding dialect vectors from ZA is slightly better than using their own. This suggests in future pooling similar dialects with less data may give better performance.

One evidence that the model successfully learns dialect-specific lexicons is "color" in US *vs.* "colour" in GB. On the GB test set, the system without any explicit dialect information (`S1`) and the one feeding it only to encoder layers (`S5`) generate recognition hypotheses with both "color" and "colour" although "color" appears much less frequently. However, for the model `S6`, where the dialect information is directly fed into decoder layers, only "colour" appears; moreover, if we feed in the dialect vector for US to `S6` on the GB test set, the model successfully switches all the "colour" predictions to "color". Similar observations are found for "labor" *vs.* "labour", "center" *vs.* "centre" etc.

Next, we feed the 1-hot dialect vector into all the layers of the LAS model (`S7`). Experimental results (Table 4) show that this system outperforms the dialect-dependent models on all the test sets, with the largest gains on AU (16.5% relative WER reduction).

4.2.3. Results using Information as Cluster Coefficients

In the literature, instead of directly feeding the dialect vector as inputs to learn a simple bias, it can also be used as a cluster coefficient vector to combine multiple clusters and learn more complex mapping functions. For comparisons, we implement a simple CAT system (`S8`) only for the encoder. Experimental results in Table 5 show that unlike directly feeding dialect vectors as inputs, CAT favors more learned embeddings (`S8(emb)`), which encourages more parameter sharing across dialects. In addition, comparing this to directly using dialect vectors (`S5(1hot)`) for the encoder, CAT (`S8(emb)`) is more effective on US and IN and similar on other dialects. However, in terms of model size, comparing to the baseline model (`S1`), `S5(1hot)` only increases by 160K parameters, while `S8(emb)` adds around 3M extra. We will leave a more thorough study of CAT for future work.

Table 5: WER (%) of a CAT encoder LAS system (`S8`) with 1-hot (`1hot`) and learned embedding (`emb`) dialect vector.

Dialect	US	IN	GB	ZA	AU	NG	KE
`S2`	9.7	16.2	12.7	11.0	12.1	33.4	19.0
`S5(1hot)`	9.6	16.4	11.8	10.6	10.7	**31.6**	**18.1**
`S8(1hot)`	9.9	17.0	12.1	11.0	11.6	32.5	18.3
`S8(emb)`	**9.4**	**16.1**	**11.7**	**10.6**	**10.6**	32.9	**18.1**

4.3. Combining Adaptation Strategies

Lastly, we integrate the joint dialect identification (`S4`) and the use of dialect vectors (`S7(1hot)`) into a single system (`S9`). The performance of this combined multi-dialect LAS system is presented in Table 6. It works much better than doing joint dialect identification (`S4`) alone, but has similar performance to the one uses dialect vectors (`S7(1hot)`). This is because when feeding in dialect vectors into the LAS model, especially in the decoder layers, the model is already doing a very good job in predicting the dialect. Specifically, the dialect prediction error for `S9` on the dev set during training is less than 0.001% compared to `S4`'s 5%. Overall, our best multi-dialect system (`S7(1hot)`) outperforms dialect-specific models and achieves 3.1~16.5% WER reductions across dialects.

Table 6: WER (%) of the combined multi-dialect LAS system (`S9`).

Dialect	US	IN	GB	ZA	AU	NG	KE
`S2`	9.7	16.2	12.7	11.0	12.1	33.4	19.0
`S4`	9.4	16.5	11.6	11.0	11.9	32.0	17.9
`S7(1hot)`	**9.1**	**15.7**	11.5	10.0	**10.1**	**31.3**	**17.4**
`S9`	**9.1**	16.0	**11.4**	**9.9**	10.3	31.4	17.5

5. CONCLUSIONS

In this study, we explored a multi-dialect end-to-end LAS system trained on 7 English dialects. The model utilizes a 1-hot dialect vector at each layer of the LAS encoder and decoder to learn dialect-specific biases. It is optimized to predict the grapheme sequence appended with the dialect name as the last symbol, which effectively forces the model to learn shared hidden representations that are suitable for both grapheme prediction and dialect classification. Experimental results show that feeding a 1-hot dialect vector is very effective in boosting the performance of a multi-dialect LAS system, and allows it to outperform a LAS model trained on each individual language. Furthermore, we also find that using CAT could potentially be more powerful in modeling dialect variations though at a cost of increased parameters, which will be addressed in future work.

6. REFERENCES

[1] W. Chan, N. Jaitly, Q. V. Le, and O. Vinyals, "Listen, Attend and Spell," *CoRR*, vol. abs/1508.01211, 2015.

[2] Fadi Biadsy, Pedro J Moreno, and Martin Jansche, "Google's cross-dialect Arabic voice search," in *Proc. ICASSP*. IEEE, 2012.

[3] Mohamed Elfeky, Meysam Bastani, Xavier Velez, Pedro Moreno, and Austin Waters, "Towards acoustic model unification across dialects," in *Proc. SLT*. IEEE, 2016.

[4] Dirk Van Compernolle, "Recognizing speech of goats, wolves, sheep and non-natives," *Speech Communication*, vol. 35, no. 1, pp. 71–79, 2001.

[5] Mohamed Elfeky, Pedro Moreno, and Victor Soto, "Multi-dialectical languages effect on speech recognition: Too much choice can hurt," in *Proc. ICNLSP*, 2015.

[6] Hui Lin, Li Deng, Dong Yu, Yi-fan Gong, Alex Acero, and Chin-Hui Lee, "A study on multilingual acoustic modeling for large vocabulary ASR," in *Proc. ICASSP*. IEEE, 2009.

[7] Zhirong Wang, Umut Topkara, Tanja Schultz, and Alex Waibel, "Towards universal speech recognition," in *Proc. ICMI*. IEEE, 2002.

[8] Hui Lin, Li Deng, Jasha Droppo, Dong Yu, and Alex Acero, "Learning methods in multilingual speech recognition," *Proc. NIPS*, 2008.

[9] Ngoc Thang Vu, David Imseng, Daniel Povey, Petr Motlicek, Tanja Schultz, and Hervé Bourlard, "Multilingual deep neural network based acoustic modeling for rapid language adaptation," in *Proc. ICASSP*. IEEE, 2014.

[10] Samuel Thomas, Sriram Ganapathy, and Hynek Hermansky, "Cross-lingual and multi-stream posterior features for low resource LVCSR systems," in *Proc. INTERSPEECH*, 2010.

[11] Georg Heigold, Vincent Vanhoucke, Alan Senior, Patrick Nguyen, M Ranzato, Matthieu Devin, and Jeffrey Dean, "Multilingual acoustic models using distributed deep neural networks," in *Proc. ICASSP*. IEEE, 2013.

[12] Kanishka Rao and Haşim Sak, "Multi-accent speech recognition with hierarchical grapheme based models," in *Proc. ICASSP*. IEEE, 2017.

[13] William Byrne, Peter Beyerlein, Juan M Huerta, Sanjeev Khudanpur, Bhaskara Marthi, John Morgan, Nino Peterek, Joe Picone, Dimitra Vergyri, and T Wang, "Towards language independent acoustic modeling," in *Proc. ICASSP*. IEEE, 2000.

[14] Arnab Ghoshal, Pawel Swietojanski, and Steve Renals, "Multilingual training of deep neural networks," in *Proc. ICASSP*. IEEE, 2013.

[15] Alex Graves, "Sequence transduction with recurrent neural networks," *arXiv:1211.3711*, 2012.

[16] Alex Graves and Navdeep Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proc. ICML*, 2014.

[17] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," in *Proc. NIPS*, 2015.

[18] Yu Zhang, William Chan, and Navdeep Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *Proc. ICASSP*. IEEE, 2017.

[19] Liang Lu, Xingxing Zhang, and Steve Renais, "On training the recurrent neural network encoder-decoder for large vocabulary end-to-end speech recognition," in *Proc. ICASSP*. IEEE, 2016.

[20] Rohit Prabhavalkar, Kanishka Rao, Tara N Sainath, Bo Li, Leif Johnson, and Navdeep Jaitly, "A Comparison of Sequence-to-Sequence Models for Speech Recognition," in *Proc. Interspeech*, 2017.

[21] Stephan Kanthak and Hermann Ney, "Multilingual acoustic modeling using graphemes," in *Proc. INTERSPEECH*. ISCA, 2003.

[22] Vassilios Diakoloukas, Vassilios Digalakis, Leonardo Neumeyer, and Jaan Kaja, "Development of dialect-specific speech recognizers using adaptation methods," in *Proc. ICASSP*. IEEE, 1997.

[23] Yan Huang, Dong Yu, Chaojun Liu, and Yifan Gong, "Multi-accent deep neural network acoustic model with accent-specific top layer using the KLD-regularized model adaptation," in *Proc. INTERSPEECH*, 2014.

[24] Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al., "Google's multilingual neural machine translation system: enabling zero-shot translation," *arXiv:1611.04558*, 2016.

[25] Tian Tan, Yanmin Qian, and Kai Yu, "Cluster adaptive training for deep neural network based acoustic model," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 3, pp. 459–468, 2016.

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

[27] Michael L Seltzer and Jasha Droppo, "Multi-task learning in deep neural networks for improved phoneme recognition," in *Proc. ICASSP*. IEEE, 2013.

[28] Ossama Abdel-Hamid and Hui Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *Proc. ICASSP*. IEEE, 2013.

[29] Chanwoo Kim, Ananya Misra, Kean Chin, Thad Hughes, Arun Narayanan, Tara Sainath, and Michiel Bacchiani, "Generation of large-scale simulated utterances in virtual rooms to train deep-neural networks for far-field speech recognition in Google Home," in *Proc. INTERSPEECH*. ISCA, 2017.

[30] Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," *arXiv:1507.06947*, 2015.

[31] Golan Pundak and Tara N Sainath, "Lower Frame Rate Neural Network Acoustic Models," in *Proc. INTERSPEECH*, 2016.

[32] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Andrew Senior, Paul Tucker, Ke Yang, Quoc V Le, et al., "Large scale distributed deep networks," in *Proc. NIPS*, 2012.

[33] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al., "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv:1603.04467*, 2016.