Data-pooling and multi-task learning for enhanced performance of speech recognition systems in multiple low resourced languages

Madhavaraj A and A G Ramakrishnan
MILE Lab, Electrical Engineering, Indian Institute of Science, Bangalore 560012
madhavaraja@iisc.ac.in, agr@iisc.ac.in

***Abstract*—We present two approaches to improve the performance of automatic speech recognition (ASR) systems for Gujarati, Tamil and Telugu. In the first approach using data-pooling with phone mapping (DP-PM), a deep neural network (DNN) is trained to predict the senones for the target language; then we use the feature vectors and their alignments from other source languages to map the phones from the source to the target language. The lexicons of the source languages are then modified using this phone mapping and an ASR system for the target language is trained using both the target and the modified source data. This DP-PM approach gives relative improvements in word error rates (WER) of 5.1% for Gujarati, 3.1% for Tamil and 3.4% for Telugu, over the corresponding baseline figures. In the second approach using multi-task DNN (MT-DNN) modeling, we use feature vectors from all the languages and train a DNN with three output layers, each predicting the senones of one of the languages. Objective functions of the output layers are modified such that during training, only those DNN layers responsible for predicting the senones of a language are updated, if the feature vector belongs to that language. This MT-DNN approach achieves relative improvements in WER of 5.7%, 3.3% and 5.2% for Gujarati, Tamil and Telugu, respectively.**

Index Terms: Multi-task learning, data-pooling, deep neural networks, phone mapping, alignments, senone posteriors, cross-lingual training, multilingual training, parameter sharing, speech recognition, Gujarati, Tamil, Telugu.

I. Introduction

Building a large vocabulary, continuous speech recognition system requires a huge corpus of transcribed speech so as to effectively estimate the acoustic model parameters, and a huge corpus of text in order to estimate the language model parameters. Although such corpora exist for English and a few other languages, there are many languages for which corpora are not readily available, and collecting such data is a cumbersome and time-consuming task. For such low-resourced languages, the traditional way of acoustic modeling results in a high word error rate. Assuming there exists similarity in acoustic units across languages, it is possible to use data from a high-resourced language in order to estimate acoustic models for a low-resourced target language [1]. In this work, we focus only on the acoustic modeling of the target language by borrowing transcribed speech corpora from one or more source languages.

Lal and King [2] have used tandem features in a cross-lingual training setting, where a neural network is trained across several languages to predict articulatory features and the outputs from this neural network are used as features to train the hidden Markov model (HMM) based acoustic models. Lu et al. [3] have used subspace Gaussian mixture models (SGMM) to learn global parameters from multiple languages and the state-specific parameters are learned from the target language data. They have also experimented maximum *a posteriori* adaptation to reduce the mismatch between the source and the target languages' SGMM global parameters. They have also extended SGMM-based cross-lingual training with l_1-regularization for estimating the state vectors [4]. This is shown to provide less word error rate, while also overcoming the problem of numerical instability. Schultz and Weibel [5] have built a language-independent speech recognition system by combining acoustic models from multiple source languages using language-separate, language-mixed and language-tagged combining methods. Manohar et al. [6] have used phone-cluster adaptive training to obtain the acoustic model parameters by linear combination of a canonical Gaussian mixture model. The mean vectors of the Gaussian mixture models for each state are parameterized by a state-vector, which is estimated through the procedure proposed by Gales [7].

Miao et al. [8] show the advantage of using deep maxout networks (DMN) in acoustic modeling. DMNs, which possess the property of dropout, have shown very good performance, particularly for low-resource languages. Sahraeian and Compernolle [9] have used manifold learning technique to derive a non-linear feature transformation from filter-bank space to articulatory space. They have used intrinsic spectral analysis and deep neural networks (DNNs) to convert acoustic features to articulatory features and used them in cross and multi-lingual training settings. Mohan and Rose [10] have used multi-task deep neural networks along with low-rank matrix factorization of the weight matrices for multi-lingual speech recognition systems. They have obtained a reduction of 44% in the number of parameters without compromising much on the word error rate (WER), when the DNNs are trained only on one hour of target language data. Heigold et al. [11] present an empirical comparison on mono-, multi- and cross-lingual training of deep neural networks for eleven languages with a

total data of 10k hours in a distributed manner. They have also shown that performing multilingual training on top of cross-lingual training gives an additional relative reduction of 5% in the WER. Data pooling of closely related languages [1] has resulted in improvements in the performance of automatic speech recognition (ASR) systems for under-resourced languages. They have shown that having two hours of data from a closely related non-target language is equivalent to having one hour of target language data.

In the work reported here, we propose two approaches to improve the performance of automatic speech recognition systems for Gujarati, Tamil and Telugu. The motivation to utilize the speech data from all the three languages to build ASR for any one of the languages arises from the similarity in the phonology of Indian languages [12]. 85% of the phones are common among Tamil, Telugu and Gujarati. Hence, for building an ASR for one of these languages as the target language, we can leverage the acoustic information from the other two languages also, for better modeling of the senone distributions. Towards this purpose, we have considered two distinct approaches, wherein the acoustic information is captured at the level of the (i) data (data pooling with phone mapping) or (ii) model (multi-task DNN).

The rest of this paper is organized as follows: In section 2, we describe our first approach of training and using a DNN to automatically map phones from source language(s) to a target language and then pooling all the source and target data together to build the speech recognition system. Section 3 describes the procedure to train a multi-task DNN using data from all the languages to predict the senones of all the languages and how the objective function is changed such that the weights are updated in a specific manner so that the first few layers capture the common acoustic information across all the languages. In section 4, we discuss the baseline system and the systems developed based on the approaches described in sections 2 and 3, and provide their results. Finally, in section 5, we conclude the paper and indicate a possible future research direction for this problem.

II. Data pooling with phone mapping

In the literature, data pooling has been used with an universal phone set [13], for cross-lingual training. However, in our approach of data pooling with phone mapping (DP-PM), we map the phones of the source languages to those of the target language using a deep neural network, trained only on the target language data. We then use this map to modify the phonetic transcriptions of the source languages to suit the target language and train the speech recognition system by all the data pooled together and fine-tune the DNN for the target language. To our knowledge, data pooling has not been used in this manner earlier. The steps involved are illustrated below. In all our experiments, the lexicon was designed by us by incorporating the pronunciation rules for the languages [14], [15]. For more details, refer [16],[17].

A. DNN training for the target language

We have used the standard procedure for training as given in s5 WSJ Kaldi recipe [18]. First we extract 39-dimensional features from the target data, consisting of mel-frequency cepstral coefficients (MFCC), delta and delta-delta features, and train a monophone model for each phone in the target language (with a total of 1000 Gaussian densities) for 40 iterations. Using the alignments from the monophone model, we then build *tri*1 models, which are triphone, context-dependent HMM models with a total of 2500 states and 15000 Gaussian densities. From the alignments of *tri*1 model, we then train *tri*2 HMM model, which is based on a combination of linear-discriminant analysis (LDA) and maximum likelihood linear transformation (MLLT). From the alignments from *tri*2, we train *tri*3, a speaker-adaptive model (LDA-MLLT-SAT) and finally we align the data using *tri*3 model. Using the probability density function (pdf) indices (also referred to as senones) from these alignments as desired target labels, and spliced MFCC features (with 5 each of left- and right-contexts) as input features, we train a *tridnn* model which is a 7-layer DNN with 2048 hidden sigmoid activations in each layer. The weights are randomly initialized and trained for 15 epochs. The learning rate is set as 0.008 for the first 4 epochs and is halved for each subsequent epoch. This DNN is now able to predict the posterior probability of a HMM state's pdf, given any input feature vector.

B. Generating alignments for the source languages

We now train the *tri*3 models for the source languages independently using the procedure explained in the previous section and then with respect to this *tri*3 model, we align the source data (also referred to as source alignments).

It is to be noted that the HMMs for the source and target languages are trained with their respective phone-sets. In order to pool all the data together, we need the data from all the languages to have a common phonetic transcription with respect to the target language's phone-set. We explain in the next section as to how we use the DNN to convert the phonetic transcription from any source language to a particular target language.

C. Mapping of phones from the source to target language

We propose this approach based on the assumption that acoustic similarities exist across languages and the function that maps such a similarity can be extracted in a data-driven fashion. We use the DNN, trained as explained in section II.A, to map the phones from any source language to the target language. Let x_s and y_s be a feature vector and its corresponding senone-id of the source language. Let $g_s(\cdot)$, $g_t(\cdot)$ be the functions that map senones to phones for the source and the target languages, respectively, and $f(\cdot)$ be the non-linear function representing the prediction of the DNN.

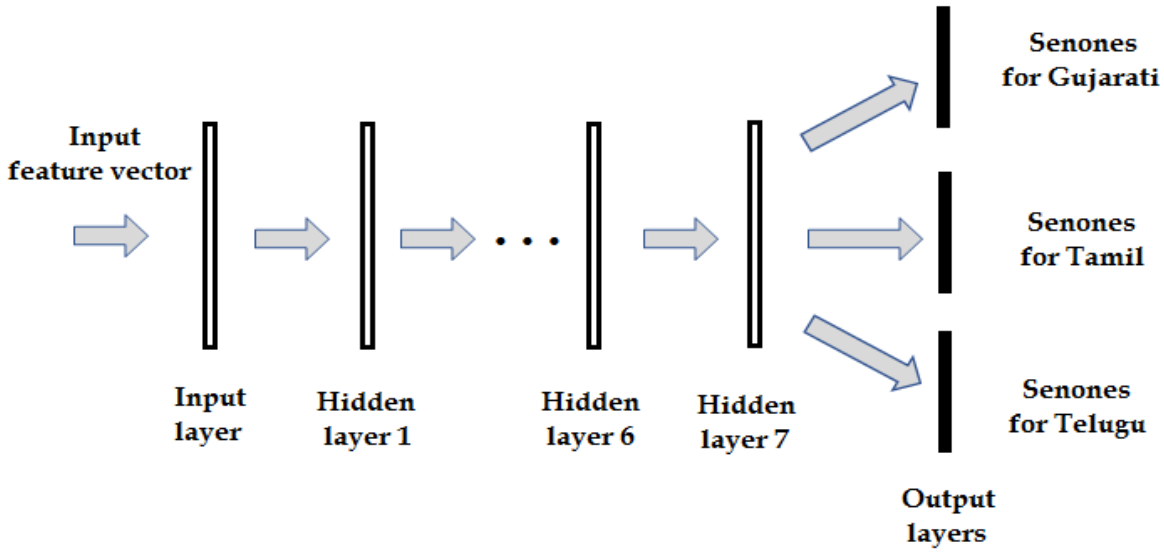

Fig. 1. Multi-task deep neural network architecture to predict senones for the three languages for any given input vector. The shared hidden layers learn feature representations common to all the languages and the three output layers perform the classification for the individual languages.

We pass all the feature vectors of the source language to the DNN and predict the senone values (denoted by $f(x_s)$). From these predictions and the senone-to-phone mapping functions, we calculate the conditional density $P(g_t(f(x_s))|g_s(y_s))$. This density function P can be calculated by counting (and then normalizing) the number of times the feature vectors belonging to a phone in the source language are recognized by the DNN as any other phone in the target language. The final function $m(\cdot)$, which maps a source language phone (ϕ_s) to a target language phone (ϕ_t^*) is given by,

$$m(\phi_s) = \phi_t^* = \underset{\phi_t}{\arg\max}\ P(\phi_t|\phi_s)$$

Using the function $m(\cdot)$, we can modify the lexicon and the phonetic transcription of any source language so that it has only the phones from the phone set of the target language.

D. Data-pooling and training

Once the source data is mapped to the format required by the target language, we pool the utterances from the source as well as the target languages and train an ASR system starting from *mono* to *tridnn* as described in section II.A.

E. Fine-tuning the DNN for the target language

At the end of the previous step, we get a DNN trained with the pooled data. Now, we fine-tune this DNN by training with only the target language data with a learning rate of 0.0008 for 5 epochs. This fine-tuned DNN can now be used as the acoustic model for decoding the test utterances of the target language.

III. Multi-task DNN approach

In our second approach, we have used a multi-task deep neural network (MT-DNN) with multiple output layers (one for each language), as shown in Fig. 1. This architecture is similar to the one in [19]. The procedure involved in training and using such an MT-DNN as the acoustic model is described below.

A. Generating alignments for the source and the target languages

We use the procedure illustrated in section II.A and build the system from *mono* to *tri*3 and using the *tri*3 models, we generate the alignments for each language independently. The number of senones may differ from language to language and thus the senones do not have a straightforward correspondence between any two languages. We learn the senone correspondence through the above MT-DNN architecture, by modifying the training procedure as explained in the subsections below.

B. Features and targets for training the MT-DNN

Let x be a feature vector and y, its corresponding senone target. The feature vector can come from any one of, say, L languages. Each training example to the MT-DNN should be of the form $\{x, [y_1, y_2, ..., y_L]\}$, where y_l is the desired target (in one-hot vector encoding format) for the l^{th} output layer. The i^{th} entry of the vector y_l is defined as,

$$y_l^i = \mathbb{1}(x \in l)\mathbb{1}(y = i) \quad \forall 1 \leq l \leq L$$

where $\mathbb{1}(\cdot)$ is the indicator function.

C. Modified loss function for training the MT-DNN

Normally, a feature vector belonging to a particular language is assigned zero as the desired target for all the other languages [11]. However, in the context of MT-DNN, since acoustic similarities exist across the languages, it is inappropriate to force the MT-DNN to predict zeros as senone-posteriors for the other languages. We handle this issue by modifying the loss function in such a way that we update only those layers responsible for predicting the senones for the language to which the feature vector belongs. This modified loss function for the l^{th} layer $\hat{L}^l$ is given by,

$$\hat{L}^l(z_l, y_l) = L(z_l, y_l)\mathbb{1}(x \in l)$$

where z_l is the actual predicted vector, y_l is the desired target vector at the l^{th} layer and $L(\cdot)$ is cross-entropy loss function. We now train the MT-DNN using this modified loss function with the feature vectors (in the format specified in section 3.2) from all the languages. We have used Keras [20] library to train this MT-DNN for 15 epochs and ported the trained network back to Kaldi format for the fine-tuning process. The learning rate was fixed at 0.008 for the first 4 epochs and reduced by half for the subsequent epochs. The main reason to train such an architecture is to ensure that the layers 1 through 7 learn feature representations that are common to all the languages and at the same time, increase the discriminability of every output layer.

Only the layers up to layer 7 learn the common representation, whereas the output layers do not learn any common representation. In other words, when a feature vector comes from a particular language, only the output layer corresponding to that language is updated. In order for the output layers to benefit from this training method, we can set the desired targets for the non-target output layers to be a predefined posterior vector instead of zeros. In such a case, there is no need to modify the loss function for training and all the output layers can be allowed to update for feature vectors from any of the languages. We hypothesize that this training procedure will further increase the performance of MT-DNN, which is yet to be experimented.

D. Fine-tuning the MT-DNN for the target language

Once the MT-DNN is trained, we retain only those layers of the MT-DNN that predict the output for the target language desired and remove the rest of the layers, thus having only one output layer. Now, we fine-tune this network for 5 epochs by using data from only the desired target language with a learning rate of 0.0008. This network can then be used as the acoustic model for decoding.

The advantage of using MT-DNN over DP-PM method is that there is no need to train the entire model set from *mono* to *tri*3 once again. It is sufficient to fine-tune the DNN only for the desired target language and use it directly for testing.

IV. Experiments and results

All our experiments have been conducted on the transcribed speech corpus given by Microsoft [21]. The training data consists of transcribed speech corpus of 40 hours for training, 5 hours for validation and 4.2 hours for testing, for each of the three languages, namely Gujarati, Tamil and Telugu. We have created the trigram language models using only the training data‘s text corpora. The CMU Indic frontend lexicon provided for each language has been used as the pronunciation dictionary. The acoustic models for the baseline systems have been built as per the procedure explained in section II.A.

Based independently on (i) data-pooling with phone mapping and (ii) multi-task DNN approaches, we have built two systems as per the procedures illustrated in sections 2 and 3, respectively. Table 1 compares the word error rates of the acoustic model (AM) of the baseline DNNs with respect to the AMs of the DNNs obtained by the two proposed methods.

Table 1 reveals that for the validation datasets, the DP-PM method gives relative improvements in WER of 1.3% for Gujarati, 1.6% for Tamil and 2.3% for Telugu. On the other hand, MT-DNN provides the best relative improvements of 3.9% for Gujarati, 1.7% for Tamil and 4.1% for Telugu.

The same trend can be seen for the blind test data as well. The DP-PM model achieves relative improvements of 5.1%, 3.1% and 3.4% over the baseline in the word error rates for Gujarati, Tamil and Telugu, respectively. The MT-DNN model results in a marginal improvement and the relative improvements achieved over the baseline are 5.7%, 3.3% and 5.2%, respectively.

Thus, our best performing MT-DNN based method gives the lowest WERs of 24.3%, 32.0% and 30.2% on the blind test data for Gujarati, Tamil and Telugu languages, respectively. We can further reduce the error rates on the test data by using both the training and the validation datasets for building the acoustic and language models.

V. Conclusion and future work

We have followed two approaches, namely data-pooling with phone mapping and multi-task DNN for cross-lingual training of the ASR for Gujarati, Tamil and Telugu languages. The first approach pools the data together by mapping the phones from the source languages to the target language, which gives relative improvements of 5.1%, 3.1% and 3.4% in the WERs for Gujarati, Tamil and Telugu test datasets, respectively. The second approach involves learning DNN model parameters from the pooled data using multi-task learning technique with a modified loss function. This achieves relative improvements in the WERs of 5.7%, 3.3% and 5.2% for Gujarati, Tamil and Telugu, respectively.

TABLE I

COMPARISON OF WERS FOR THE BASELINE, DATA-POOLING WITH PHONE MAPPING (DP-PM) AND MULTI-TASK DNN (MT-DNN) MODELS ON VALIDATION AND TEST SETS. RELATIVE IMPROVEMENT IN WER (IN %) WITH RESPECT TO THE BASELINE IS GIVEN IN PARENTHESES FOR EACH CASE.

Method	**Gujarati**		**Tamil**		**Telugu**	
	Val. set	Test set	Val. set	Test set	Val. set	Test set
Baseline	18.8 (NA)	25.7 (NA)	32.8 (NA)	33.1 (NA)	30.6 (NA)	31.9 (NA)
DP-PM	18.6 (1.3)	24.4 (5.1)	32.3 (1.6)	32.1 (3.1)	29.9 (2.3)	30.8 (3.4)
MT-DNN	18.1 (3.9)	24.3 (5.7)	31.3 (1.7)	32.0 (3.3)	29.3 (4.1)	30.2 (5.2)

Our future work will involve extending the multi-task learning approach by using mean statistic of the senone-posterior outputs for the feature vectors belonging to a particular senone class as desired targets for the MT-DNN, instead of predicting zeros for the non-target languages, without modifying the loss function.

REFERENCES

[1] C. Van Heerden, N. Kleynhans, E. Barnard, and M. Davel, "Pooling ASR data for closely related languages," in *SLTU 2010: Proc. 2nd Workshop on Spoken Languages Technologies for Under-resourced languages*, 2010, pp. 17–23.

[2] P. Lal and S. King, "Cross-lingual automatic speech recognition using tandem features," *IEEE Trans. Audio, Speech, and Language Processing*, vol. 21, no. 12, pp. 2506–2515, 2013.

[3] L. Lu, A. Ghoshal, and S. Renals, "Cross-lingual subspace gaussian mixture models for low-resource speech recognition," *IEEE/ACM Trans. Audio, Speech and Language Processing (TASLP)*, vol. 22, no. 1, pp. 17–27, 2014.

[4] ——, "Regularized subspace gaussian mixture models for cross-lingual speech recognition," in *Automatic Speech Recognition and Understanding (ASRU), IEEE Workshop on*, 2011, pp. 365–370.

[5] T. Schultz and A. Waibel, "Language-independent and language-adaptive acoustic modeling for speech recognition," *Speech Communication*, vol. 35, no. 1-2, pp. 31–51, 2001.

[6] V. Manohar, C. B. Srinivas, and S. Umesh, "Acoustic modeling using transform-based phone-cluster adaptive training," in *Automatic Speech Recognition and Understanding (ASRU), IEEE Workshop on*, 2013, pp. 49–54.

[7] M. J. F. Gales, "Cluster adaptive training of hidden Markov models," *IEEE Trans. Speech and Audio Processing*, vol. 8, no. 4, pp. 417–428, 2000.

[8] Y. Miao, F. Metze, and S. Rawat, "Deep maxout networks for low-resource speech recognition," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, Dec 2013, pp. 398–403.

[9] R. Sahraeian and D. V. Compernolle, "Crosslingual and multilingual speech recognition based on the speech manifold," *IEEE/ACM Trans. Audio, Speech, and Language Processing*, vol. 25, no. 12, pp. 2301–2312, 2017.

[10] A. Mohan and R. Rose, "Multi-lingual speech recognition with low-rank multi-task deep neural networks," in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conf. on*, 2015, pp. 4994–4998.

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

[12] K. V. Vijay Girish, V. Veena, and A. G. Ramakrishnan, "Relationship between spoken Indian languages by clustering of long distance bigram features of speech," in *India Conference (INDICON), 2016 IEEE Annual*. IEEE, 2016, pp. 1–6.

[13] J. L. Hieronymus, "ASCII phonetic symbols for the world's languages: Worldbet," 1994.

[14] A. G. Ramakrishnan and M. Laxmi Narayana, "Grapheme to phoneme conversion for Tamil speech synthesis," in *Proc. Workshop in Image and Signal Processing (WISP-2007), IIT Guwahati*, 2007, pp. 96–99.

[15] A. G. Ramakrishnan, R. D. Sequiera, S. S. Rao, and H. R. Shiva Kumar, "Transliteration of Indic languages to Kannada with a user-friendly interface," in *Advance Computing Conference (IACC), 2015 IEEE International*. IEEE, 2015, pp. 998–1001.

[16] A. Madhavaraj and A. G. Ramakrishnan, "Design and development of a large vocabulary, continuous speech recognition system for Tamil," in *2017 14th IEEE India Council International Conference (INDICON)*. IEEE, 2017, pp. 1–5.

[17] A. Madhavaraj, H. R. Shiva Kumar, and A. G. Ramakrishnan, "Online speech translation system for Tamil," in *19th Annual Conference of International Speech Communication Association (Interspeech 2018)*. IEEE, 2018.

[18] 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 *Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[19] 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," in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conf. on*, 2013, pp. 7304–7308.

[20] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015.

[21] *Data provided by SpeechOcean.com and Microsoft*. Microsoft, 2018.