EXPLOITING SEQUENTIAL LOW-RANK FACTORIZATION FOR MULTILINGUAL DNNS

Reza Sahraeian, Dirk Van Compernolle

KU Leuven - ESAT, Kasteelpark Arenberg 10, 3001 Heverlee, Belgium

{Reza.Sahraeian,Dirk.VanCompernolle}@esat.kuleuven.be

ABSTRACT

DNNs have shown remarkable performance in multilingual scenarios; however, these models are often too large in size that adaptation to a target language with relatively small amount of data cannot be well accomplished. In our previous work, we utilized Low-Rank Factorization (LRF) using singular value decomposition for multilingual DNNs to learn compact models which can be adapted more successfully. In this paper, we address two problems associated with that LRF scheme and we propose a compellingly simple methodology to overcome them. First, factorizing all layers results in a huge drop in performance and consequently a long recovery process is required which is not practically efficient. Secondly, LRF can be viewed as a regularization by which some noise is added to weight layers; however, factorizing all layers together equates to adding too much noise which results in bad performance. To mitigate these problems, we propose to apply LRF sequentially. We demonstrate that the lost information after factorizing one layer is small and can be rapidly retrieved; hence, sequential factorization is more efficient. Moreover, the sequential LRF adds only a small amount of noise sequentially which is a better regularization. Our experiments are conducted on five languages from the GlobalPhone dataset.

Index Terms— Multilingual DNNs, low-rank factorization

1. INTRODUCTION

In recent years, DNNs have been used in a large body of research to exploit out-of-language data particularly for under-resourced speech recognition [1, 2, 3, 4, 5]. The key assumption in multilingual DNNs is that the hidden layers can be considered as a universal complex feature transformation and can be shared across languages while the softmax layers are language dependent [1, 6]. This suggests that the hidden layers can be trained simultaneously for different languages to benefit from each other. Furthermore, for a specific target language, it has been shown that additional gain over the purely multilingual DNN can be obtained by adapting the multilingual DNN using data from the target language [7].

The trend in DNNs is to increase the number of parameters to fully exploit ever-increasing training data [8]. This, however, entails long training time and slow prediction; and furthermore, once DNNs are being deployed on mobile and embedded devices with little memory, such large models cannot be stored. This dilemma motivates several studies recently to investigate DNN size reduction without hurting the performance. As examples, [9] proposed a parameter sharing scheme using a hash function and a drastic reduction in model size was achieved. Another work proposed to sparsify the weight matrices by employing regularizers [10]. Another popular approach in this area is low-rank matrix factorization [11, 12, 13]; the core idea is to represent the weight matrix as a low-rank product of two smaller matrices. Low-rank factorization (LRF) can be employed either with a linear bottleneck [11] or singular value decomposition (SVD) [12]. Very recently, LRF and parameter sharing scheme have been effectively used to reduce the number of parameters of a standard LSTM by 75% at a small cost of 0.3% increase in WER [14].

While the main intent of the aforementioned studies is to reduce the model size and accelerate the DNN training and test time, no significant improvement is achieved. However, we have shown in our previous works that the model size reduction via LRF in a multilingual DNN provides significant gain over a conventional multilingual DNN [15, 16]. This improvement is mainly achieved in the adaptation phase where much smaller number of parameters need to be adapted to a specific target language with relatively small amount of data. Moreover, it has been shown that factorizing only the final weight layer is beneficial in a multilingual DNN with shared hidden layers which can be due to the fact that the number of parameters which are trained with language specific data is reduced [15, 17].

Despite the gain obtained from LRF of multilingual DNN, there still remain some issues with respect to efficiency and accuracy. In [15], we proposed to apply SVD to all hidden layers at the same time; this normally results in a huge model size reduction and consequently the multilingual DNN moves away from its optimal trained state. To remedy this problem, as suggested in [12], we need to retrain the whole network. However, depending on how far the factorized model is moved from the original DNN, we may require a long retraining process to gain back the lost information as much as possible. Moreover, the performance drop in the factorized DNN might never be completely recouped with retraining if the model is too far from the original one [12]. We observed in [15] that even if the factorized network is not as good as the original one, when adapting the network from this starting point to a target language, we ultimately reach a better performance. Yet, by regaining more information after the LRF, a more informative factorized model will be deployed for adaptation and thus better performance might be obtained.

In this respect, the motivation of this paper is to propose an efficient framework for LRF of multilingual DNNs. Our proposed approach is compellingly simple: we deploy the LRF in a sequential manner. In other words, the SVD is applied layer by layer and after each layer is factorized, DNN is retrained with a small number of iterations only. We demonstrate that this technique is very efficient and fast in regaining the lost information from the compressed DNN compared to the conventional LRF. Moreover, by viewing LRF as a regularizer of the model space which can improve the performance [18], factorizing all layers together may degrade the DNN performance by adding a large amount of noise while the sequential LRF adds small amounts of noise sequentially which we believe is a better generalization scheme. The rest of the paper is organized as follows. In section 2, we describe the LRF and sequential LRF for multilingual DNNs. The experimental setup and results are presented in sections 3 and 4. Finally, we have concluding remarks.

2. LRF IN MULTILINGUAL DNNS

2.1. Multilingual DNNs

A conventional multilingual DNN with shared hidden layers has been formulated in the multi-task learning framework [2]. The hidden layers as feature extractors and the classifiers are jointly optimized on the shared data for different languages. The output layers are usually context-dependent states determined by standard clustering algorithms from previously trained HMMs [19].

Given a target language, multilingual DNN performance can be further improved by adjusting the parameters of the whole network through retraining the DNN with data of the target language which is often termed as adaptation [7]. However, the improvement is restricted to the amount of adaptation data and size of the DNN; usually, the adaptation data from the target language is relatively small and the largely parametric multilingual DNN is likely to overtrained. We have shown that the multilingual DNN adaptation benefits from LRF as the model size is reduced [15]. In the next section we briefly explain LRF for multilingual DNNs.

2.2. Low-rank factorization (LRF)

The use of low-rank matrix factorization for DNN training is proposed in [11] and [12] to reduce computational and space complexity for monolingual DNNs. To this end, each connection weight matrix can be factorized into smaller matrices and thereby the number of parameters in the network is significantly reduced. Especially when DNNs are trained with a large number of output targets, [11] shows that LRF of the last weight layer reduces the number of parameters of the DNN significantly.

Let us denote the final weight matrix for language L by A^L with dimensions $n_H \times n_T^L$ where n_H is the number of units in the last shared hidden layer and n_T^L is the number of output targets for language L. If there is a rank n_r for the weight matrix, then there exists a factorization $A^L = B^L \times C^L$ where B^L and C^L are full rank matrices of size $n_H \times n_r$ and $n_r \times n_T^L$ respectively. Now, in a multilingual low-resource scenario we may want to further reduce the number of language dependent parameters by incorporating the matrix B^L in the layers that are shared across languages and thus $B^L = B$ for all languages [17]. Then, for a language L', we only need to train an output weight matrix of dimensions $n_r \times n_T^{L'}$, which is much smaller than $n_H \times n_T^{L'}$. It is worth noting that in this approach there exists one extra weight layer in the shared components compared to the typical multilingual DNN; however, we have shown in [15] that this is not very relevant.

In our work, LRF of the weight layers is done by using SVD based model restructuring method in which a $n_H \times n_T^L$ weight matrix layer A^L is decomposed as:

$$A^L_{n_H \times n_T^L} \approx U_{n_H \times n_r} \Sigma_{n_r \times n_r} V^T_{n_T^L \times n_r} \quad (1)$$

Then, we consider $B = U_{n_H \times n_r}$ and $C^L = \Sigma_{n_r \times n_r} V^T_{n_T^L \times n_r}$ and replace A^L with these two smaller matrices as described in [12]. Furthermore, we proposed in [15] to extend LRF to other weight layers which leads to a huge reduction of the number of parameters in the multilingual DNN system. For the hidden layers, factorization is more straightforward and we simply need to factorize the weight matrix of size $n_H \times n_H$ into smaller matrices of size $n_H \times n_r$ and $n_r \times n_H$. It is also worth noting that in our LRF approach we skip factorizing the input weight layer as it is a very small matrix compared to the other weight matrices. The LRF is applied after initial training of the multilingual DNN and before adaptation to a target language data.

LRF can also be accomplished by configuring the DNN with a linear bottleneck and let the factorization being learned during DNN training, and since parameters of DNN is reduced before training, the overall training time can be reduced as well [11]. The downside of this method, however, is that the bottleneck dimension has to be defined beforehand and for a new dimensionality we need to train a new DNN. However, the main goal of this paper is to improve the accuracy rather than decreasing training time; thus, SVD is applied to factorize the weight layers so that n_r can be tuned with less computational complexity.

2.3. Sequential LRF

Although LRF in the way explained in the previous section provides a huge model size reduction, it suffers from some issues. First of all, factorizing all layers together can drastically move the network away from the local optimum that was reached during training. This necessitates a recovery stage which can be simply a retraining. However, if LRF leads to a huge reduction in number of parameters, a large amount of noise is added to the model and consequently a long retraining process is required. Not only is not this practically efficient, but also the factorized model might not be recovered if there is a huge drop in size and performance. The second issue is that LRF of DNNs in the previous studies is mostly viewed as a way to reduce the model size; however, if being employed properly, it can take a role on regularization and improves the model performance with respect to generalization. Nonetheless, we show in the experiments that factorizing all layers together causes a huge damage to the model performance and is not a good regularization method.

Being motivated by the aforementioned issues, we propose to apply LRF sequentially to the weight layers. The reasoning is that we believe factorization of only one layer adds only a small amount of noise to the weight layer and does not drastically move the model from its current state. Therefore, the chance to quickly retrieve the small lost information increases. In terms of regularization, adding small noise sequentially is a more appropriate methodology rather than imposing a large amount of noise to the model only one time.

For implementation, we start from the output weight layer; once this layer is factorized, we let the model to be retrained with the whole multilingual data for a short time. Then, one layer before the output one is factorized and again the whole model is retrained. This procedure continues until all layers are factorized (except the first input layer). If there are N samples in the multilingual training data pool which are shuffled, we can manage to retrain the DNN after each step of factorization using a subset of the training samples. For example, if there are 8 weight layers to be factorized and we retrain with $N/8$ samples after applying SVD to each layer, the whole factorization will be finished after 1 epoch. The important question is that how long each retraining stage needs to be. To answer this question, we investigated several scenarios and we successfully show that even in one epoch, the whole factorization can succeed.

3. EXPERIMENTAL SETUP

3.1. ASR systems

Monolingual reference systems were built using target language data only. We only report the monolingual results using DNNs as they outperform GMM systems. Monolingual DNNs were trained by adopting the audio alignments from the conventional HMM/GMM systems. The input features for the DNNs were mean and variance normalized 23-dimensional FBANK features being concatenated with 7 left and 7 right neighbor frames to yield an input feature

Table 1. Baseline results in WER(%) for the five languages using monolingual and multilingual systems.

Settings		Monolingual DNN	Multilingual DNN	
			Not adapted	Adapted
FR	Dev.	26.45	25.90	25.15
	Eval.	23.90	23.65	23.48
SP	Dev.	17.78	17.47	17.14
	Eval.	10.42	9.73	9.57
PO	Dev.	19.70	19.27	18.87
	Eval.	20.97	20.48	19.84
RU	Dev.	32.88	32.64	31.99
	Eval.	31.37	30.60	30.25
GE	Dev.	11.85	11.15	11.02
	Eval.	19.49	18.78	18.36

vector size of 345; we observed that FBANK features outperform MFCCs as input features for DNN.

The multilingual systems were based on multi-task learning of DNNs. The neural network's input features were the same as those used in the monolingual DNNs except that normalization was not applied. More details about the implementations are provided in the experiment section. All the DNNs used in this study were trained using a ReLU nonlinearity based on greedy layerwise supervised training [20]. The initial and final learning rates were specified by hand and equal to 0.01 and 0.001 respectively. We used the Kaldi ASR toolkit [21] in our experiments.

3.2. GlobalPhone

The GlobalPhone corpus is a multilingual text and speech corpus that covers speech data from 20 languages [22]. In this work, multilingual data consist of five languages from the GlobalPhone dataset: French (FR), Spanish (SP), Portuguese (PO), Russian (RU) and German (GE) and the available training data for these languages are 22.74hr, 22.71hr 21.10hr 17.55hr and 14.85hr respectively. The detailed statistics for these languages are presented in [22]. The recognition task is a standard word recognition task using a trigram language model obtained from Karlsruhe University[1].

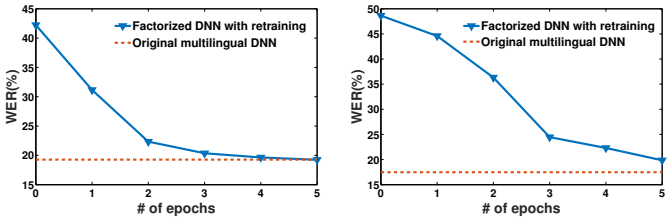
4. EXPERIMENTS

First, we constructed baseline systems for the five languages in both monolingual and multilingual fashions. Following the setup in [1], the number of target context-dependent states was set to 3100 for each language. The monolingual results for both development (Dev.) and evaluation (Eval.) sets are presented in Table 1. Also, a multilingual DNN was trained with a dedicated softmax layer for each language while the hidden and input layers were shared. We used a DNN with 8 hidden layers for the multilingual setting and the number of nodes was 1500 per layer; noting that these values were tuned. The performance of the multilingual systems with and without adaptation is presented in Table 1. From Table 1 we can observe that the multilingual DNN performs the best; moreover, adaptation yields a small improvement which is a typical behaviour for a multilingual DNN with a large number of parameters.

4.1. Low-rank factorization

We carried out sets of experiments to investigate the effectiveness and efficiency of LRF in the multilingual DNN. To this end, SVD was applied on all the weight layers except the input weight layer

[1]http://csl.ira.uka.de/GlobalPhone/



(a) Portuguese (b) Spanish

Fig. 1. Tracking the WER(%) in retraining the low-rank factorized multilingual DNN.

of the not adapted multilingual DNN, and afterwards the whole network was retrained with multilingual data. Following our setup in [15], we consider $n_r = 500$. Since in our experiment $n_H = 1500$, this factorization reduces the number of parameters in each hidden weight layer by a factor of 0.66. In total, the reduction in DNN model size for each language is around 63%; this suggests that in the adaptation phase we have much smaller number of parameters to adapt to a target language. However, the factorization degrades the performance of the multilingual DNN and in order to regain this performance we need to retrain the multilingual DNN. Figure 1 reveals how the performance of the multilingual DNN changes after factorization and during the course of retraining; the results in this figure are presented on Dev. sets for SP and PO as two examples.

The following observations can be made from Figure 1. First, the LRF degraded the performance of multilingual DNN drastically which is not surprising as a big compression had happened. However, it can be seen that during the retraining process a big part of the gap was filled. After five epochs, the performance of the factorized multilingual DNN almost equals the original intact multilingual DNN for Portuguese; for Spanish, however, there still remains a small difference after 5 epochs. We could indeed increase the number of epochs and hopefully better convergence would have been obtained for Spanish, but we should note that since retraining is applied multilingually, increasing the number of epochs might lead to overfitting for other languages. Besides, it is not efficient to have a long retraining because the choice of n_r is usually set empirically and we are interested in finding the proper value rapidly.

We then deployed the factorized multilingual DNN obtained from the previous retraining stage in the adaptation phase. Since the DNN size is reduced, the model adaptation is expected to succeed more effectively. Table 2 shows the WER for different languages using the factorized multilingual DNN before and after adaptation. The starting point for adaptation to the specific target language was the factorized model which had been retrained multilingually for 5 epochs. The following trends are apparent from Table 2. First, the factorized multilingual DNN being retrained for 5 epochs mostly performs worse than the original multilingual DNN, and for languages like Spanish and Russian the rise in WER is pronounced. However, when the factorized model is adapted to the target languages, the WERs reduced noticeably for all languages. This can be understood by the reasoning that LRF has created a network with fewer, but more relevant parameters. In this set of experiments the learning rate for adaptation to the target languages was set to 0.0001.

4.2. Sequential low-rank factorization

In this part of experiments, we investigated the effectiveness and efficiency of the sequential LRF described in section 2.3. Towards this goal, like the experiments for the conventional LRF, the original multilingual DNN without adaptation was deployed as the starting point. However, instead of factorizing all weight layers at the same

Table 2. Comparing WER(%) using LRF with and without adaptation with the adapted standard multilingual DNN.

Target languages		LRF		Baseline multilingual DNN
		Not adapted	Adapted	
FR	Dev.	26.04	25.07	25.15
	Eval.	24.25	23.40	23.48
SP	Dev.	19.85	16.37	17.14
	Eval.	10.16	9.14	9.57
PO	Dev.	19.24	18.38	18.87
	Eval.	19.75	19.47	19.84
RU	Dev.	34.42	31.32	31.99
	Eval.	33.26	29.96	30.25
GE	Dev.	11.04	10.27	11.02
	Eval.	17.74	16.79	18.36

Table 3. Comparing WER(%) on Dev. sets using conventional LRF and sequential LRF for different retraining durations.

Target languages	Retraining duration (epochs) for sequential LRF					LRF
	1	2	3	4	5	
FR	25.90	24.67	24.57	24.54	24.44	26.04
SP	18.11	16.95	16.82	16.32	16.36	19.85
PO	20.08	19.14	18.99	19.01	18.80	19.24
RU	34.61	33.39	32.99	33.08	32.93	34.42
GE	11.67	10.89	10.51	10.52	10.70	11.04

time, we applied the factorization layer by layer. Again, we set $n_r = 500$ for the sake of fair comparison to the previous experiments. At the beginning, only the final weight layer was factorized and the model was retrained for the fixed number of samples. Then, the next weight layer right before the final weight layer was factorized and again model was retrained for a while. This procedure continued until all hidden weight layers were factorized.

First, we examined different scenarios in terms of the retraining duration after each factorization. Since the multilingual DNN in our work includes eight hidden layers, factorization needed to be applied eight times (we don't factorize the input weight layer). After each factorization, the model was retrained for the specified number of training samples. Table 3 summarizes the results obtained from the sequential LRF for different retraining duration on Dev. sets of different languages. In this table, "retraining duration" refers to the number of epochs required to have all layers factorized. The last column presents the results from Table 2 where we applied LRF to all layers and retrained the model for 5 epochs.

Interesting observations are made from Table 3. First of all, it can be seen that by applying sequential LRF, the proper compressed model can be obtained in a very short retraining duration. For example in Fig. 1, we observed that five epochs of retraining with multilingual data was required to almost regain the lost information from LRF of all layers for Spanish; however, when LRF is employed sequentially in one epoch, we achieved the performance which is even closer to the original multilingual DNN. The results of applying sequential LRF in small number of epochs for different languages suggest that factorizing one individual layer led to a small reduction in model performance which was easily regained. The more interesting point being apparent in Table 3 is that in many cases sequential LRF even without adaptation improves the performance compared to the original multilingual DNN shown in Table 1. For example, the original multilingual DNN provides a WER of 17.47% on Dev. set for Spanish; taking this model as the starting point and applying sequential LRF in the duration of 4 epochs, the WER reduces to 16.32%. However, we can observe that when LRF was applied to all layers in one step, we could get the WER of 19.85% after retraining for 5 epochs. This improvement can be attributed to the fact that by factorizing only one weight layer a small amount of noise is added to the weight matrix and this can be viewed as a good generalization to the DNN. Besides, we should note that the gain by sequential LRF was achieved before adaptation and further improvement might still be obtained by adaptation of this compact model.

Table 4. WER (%) for sequential LRF with and without adaptation. Relative WERs reduction compared to the standard multilingual DNN with adaptation are also presented.

Target languages		Sequential LRF in three epochs		Relative WER (%) reduction
		Not adapted	Adapted	
FR	Dev.	24.57	**24.01**	4.5
	Eval.	23.18	**22.64**	3.6
SP	Dev.	16.82	**15.94**	7
	Eval.	9.48	**9.03**	5.6
PO	Dev.	18.99	**18.00**	4.6
	Eval.	19.52	**18.53**	6.6
RU	Dev.	32.99	**30.52**	4.6
	Eval.	31.30	**29.00**	4.1
GE	Dev.	10.51	**10.10**	8.3
	Eval.	17.10	**16.44**	10.44

Moreover, we monitored the model performance prior and after each step of factorization. We observed that in many cases the WER absolute reduction after each step of factorization is less than 1% and retraining easily regained most of it. For example, for German data we observed that in the first step by factorizing only the final weight layer, the WER on Dev. set increased from 11.04% to 11.95%; or in the 6th step, the increase in WER was only 0.56%. This confirms our hypothesis that sequential LRF leads to only small drop in model performance at each step which can be easily retrieved.

Finally, we present the results of adaptation on top of the sequentially factorized multilingual DNN. Table 4 shows that further improvements were obtained by adaptation and the results highlighted in this table are the best recognition performance we achieved. Comparing this table with Table 2 reveals that sequential LRF provided a better factorized model compared to the conventional LRF, and consequently adaptation led to higher recognition performances. In the last column, we also present the relative WER reduction obtained by using sequential LRF and adaptation compared to the standard adapted multilingual DNN.

5. CONCLUSIONS

The results of this paper show that we have successfully improved the LRF of multilingual DNN. We proposed to employ LRF in a sequential manner to deal with two major issues associated with the conventional LRF. In the experiments on five languages from the GlobalPhone dataset, we demonstrated the effectiveness of the sequential LRF. From the combined sets of experiments we may draw the following conclusions: (i) In all scenarios, using conventional LRF together with adaptation improved the recognition results. (ii) The proposed sequential LRF significantly reduces the required retraining time and even with one epoch of retraining a proper compact model can be obtained. (iii) Sequential LRF in combination with adaptation can boost the results with 3.6-10.44% relative in comparison with the normal multilingual DNN with adaptation.

6. REFERENCES

[1] Arnab Ghoshal, Pawel Swietojanski, and Steve Renals, "Multilingual training of deep neural networks," in *ICASSP*. IEEE, 2013, pp. 7319–7323.

[2] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *ICASSP*. IEEE, 2013, pp. 7304–7308.

[3] Mark JF Gales, Kate M Knill, Anton Ragni, and Shakti P Rath, "Speech recognition and keyword spotting for low resource languages: babel project research at CUED," in *Spoken Language Technologies for Under-Resourced Languages*, 2014, pp. 16–23.

[4] David Imseng, Petr Motlicek, Philip N Garner, and Hervé Bourlard, "Impact of deep MLP architecture on different acoustic modeling techniques for under-resourced speech recognition," in *ASRU*. IEEE, 2013, pp. 332–337.

[5] Frantisek Grezl, Martin Karafiát, and Milos Janda, "Study of probabilistic and bottle-neck features in multilingual environment," in *ASRU*. IEEE, 2011, pp. 359–364.

[6] Karel Veselỳ, Martin Karafiát, Frantisek Grézl, Milos Janda, and Ekaterina Egorova, "The language-independent bottleneck features," in *Workshop on Spoken Language Technology (SLT)*, 2012, pp. 336–341.

[7] Frantisek Grézl, Martin Karafiát, and Karel Vesely, "Adaptation of multilingual stacked bottle-neck neural network structure for new language," in *ICASSP*. IEEE, 2014, pp. 7654–7658.

[8] Adam Coates, Brody Huval, Tao Wang, David Wu, Bryan Catanzaro, and Ng Andrew, "Deep learning with COTS HPC systems," in *ICML*, 2013, pp. 1337–1345.

[9] Wenlin Chen, James Wilson, Stephen Tyree, Kilian Q Weinberger, and Yixin Chen, "Compressing neural networks with the hashing trick," in *ICML*, 2015, pp. 2285–2294.

[10] Maxwell D Collins and Pushmeet Kohli, "Memory bounded deep convolutional networks," *arXiv preprint arXiv:1412.1442*, 2014.

[11] Tara N Sainath, Brian Kingsbury, Vikas Sindhwani, Ebru Arisoy, and Bhuvana Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *ICASSP*. IEEE, 2013, pp. 6655–6659.

[12] Jian Xue, Jinyu Li, and Yifan Gong, "Restructuring of deep neural network acoustic models with singular value decomposition.," in *INTERSPEECH*, 2013, pp. 2365–2369.

[13] Misha Denil, Babak Shakibi, Laurent Dinh, Nando de Freitas, et al., "Predicting parameters in deep learning," in *NIPS*, 2013, pp. 2148–2156.

[14] Zhiyun Lu, Vikas Sindhwani, and Tara N Sainath, "Learning compact recurrent neural networks," in *ICASSP*. IEEE, 2016, pp. 5960–5964.

[15] Reza Sahraeian and Dirk Van Compernolle, "A study of rank-constrained multilingual dnns for low-resource asr," in *ICASSP*. IEEE, 2016, pp. 5420–5424.

[16] Reza Sahraeian and Dirk Van Compernolle, "Using weighted model averaging in distributed multilingual dnns to improve low resource ASR," *Procedia Computer Science*, vol. 81, pp. 152–158, 2016.

[17] Aanchan Mohan and Richard Rose, "Multi-lingual speech recognition with low-rank multi-task deep neural networks," in *ICASSP*. IEEE, 2015, pp. 4994–4998.

[18] Ming Yuan, Ali Ekici, Zhaosong Lu, and Renato Monteiro, "Dimension reduction and coefficient estimation in multivariate linear regression," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 69, no. 3, pp. 329–346, 2007.

[19] Hervé Bourlard, Nelson Morgan, Chuck Wooters, and Steve Renals, "CDNN: A context dependent neural network for continuous speech recognition," in *ICASSP*. IEEE, 1992, vol. 2, pp. 349–352.

[20] Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *arXiv preprint arXiv:1410.7455*, 2014.

[21] Daniel Povey et al., "The KALDI speech recognition toolkit," in *ASRU*, 2011, pp. 1–4.

[22] Tanja Schultz, Ngoc Thang Vu, and Tim Schlippe, "Globalphone: A multilingual text & speech database in 20 languages," in *ICASSP*. IEEE, 2013, pp. 8126–8130.