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

Compact Speaker Embedding: lrx-vector

Munir Georges[1,3], *Jonathan Huang*[2*], *Tobias Bocklet*[1,4]

[1]Intel Labs, Munich, Germany
[2]Apple Inc, Cupertino, California, USA
[3]Technische Hochschule Ingolstadt, Germany
[4]Technische Hochschule Nürnberg, Germany

munir.georges@intel.com, jjhuang@apple.com, tobias.bocklet@intel.com

Abstract

Deep neural networks (DNN) have recently been widely used in speaker recognition systems, achieving state-of-the-art performance on various benchmarks. The x-vector architecture is especially popular in this research community, due to its excellent performance and manageable computational complexity. In this paper, we present the lrx-vector system, which is the low-rank factorized version of the x-vector embedding network. The primary objective of this topology is to further reduce the memory requirement of the speaker recognition system. We discuss the deployment of knowledge distillation for training the lrx-vector system and compare against low-rank factorization with SVD. On the VOiCES 2019 far-field corpus we were able to reduce the weights by 28% compared to the full-rank x-vector system while keeping the recognition rate constant (1.83 % EER).
Index Terms: speaker recognition, x-vector, low power

1. Introduction

Speaker recognition systems have been popularized in consumer devices such as smart phones and smart speakers, for access control. This is achieved by generating a voice print from the user's speech during interaction with the device and comparing against an existing voice print. Voice prints are usually generated by speaker embeddings of Deep Neural Networks (DNNs), which can also be the underlying feature for diarization in multi-speaker meetings [1, 2]. DNNs have extensively explored in the literature for the generation of speaker embedding with different objective functions [3, 4, 5, 6]. The x-vector system [3] emerged as a favorite in the research community, due to its robust training, state-of-the-art performance and the availability of recipes in the popular Kaldi framework [7]. Our work here is focused on the x-vector system.

Local inference provides a clear advantages over cloud solution. Examples are: improved protection of user privacy, lower recognition latency, or the autonomy from communications channels. Local inference has been previously addressed for speech recognition [8] and spoken language understanding [9]. The main challenges in local inference is the limited compute and memory available on the device. Increasing these requirements have adverse implications to cost and energy efficiency of the device. The memory access operations during inference is identified as a bottleneck for energy efficiency. In this paper, we focus on reducing the memory footprint of an x-vector-based speaker verification system. Furthermore, reduction in memory footprint will lead to lower cost for the device.

There is already a wealth of literature available the focus on the compression of DNNs. Training DNNs with low-rank matrices jointly with the target objective is explored for vision and audio signals, previously. Novikov et al. [10] explore low-rank factorization of neural networks using CIFAR-10 and 1000-class ImageNet ILSVRC-2012.Sak et al. [11] use low-rank projection layers in Recurrent Neural Networks (RNN) for speech recognition. A rank constrained DNN topology for key word spotting is proposed by Nakkiran et al. [12]. Related work to compression, HashNet uses a low-cost hash function to randomly group weights into hash buckets. Chen et al. [13] propose and compare this approach with low-rank networks. Wu et al. [14] explore quantization of convolutional neural networks and compares them with various alternatives including Low-rank Decomposition and Approximation of Non-linear Responses. An energy-efficient hardware accelerator using a low-rank approximation is also proposed by Zhu et al. [15] where inactive neurons are passed by. Sahraeian et al. [16] explore low-rank factorization beyond compression aspects via Singular Value Decomposition (SVD) of the weight matrices to achieve sparse multilingual acoustic models. More general, Dighe et al. [17] improve the acoustic model by training using low-rank and sparse soft targets. Similar success was achieved for Deep Gaussian Conditional Random Fields as explored by, e.g., Chandra et al. [18]. Ding et al. [19] describe a structured low-rank constraint using domain-specific and domain-invariant DNNs. Applying low-rank and low-rank plus diagonal matrix parametrization to RNNs is studied by Barone et al. [20]. Sharan et al. [21] explore random projection for low-rank tensor factorization and describe the use on gene expression and EEG time series data. Zhang et al. [22] apply structural sparsification on Time-Delay Neural Networks (TDNN) to remove redundant structures. Alternative approaches are subject to our further research, e.g., binary neural networks as successfully applied to natural language understanding [23].

In this paper, we proposed low-rank x-vector speaker embeddings by deploying knowledge distillation to the training process. We call the resulting embedding lrx-vector. The paper is organizes as follows: Section 2 describes the baseline speaker recognition system, with details on the model topology and loss function. In Section 3, we introduce the modified model topology and the model training methodology using knowledge distillation. We present the results of our experiments in Section 4 and then conclude the paper in 5.

2. Baseline Speaker Embedding System

The x-vector embedding comprises two parts. First, the feature sequence, i.e., mel-filter bank is processed by layers of TDNNs. Second, a statistical pooling layer encodes a segment of speech and computes the embedding vector by a feed forward network. This architecture is illustrated in Figure 1.

*Work done at Intel Labs

http://dx.doi.org/10.21437/Interspeech.2020-2106

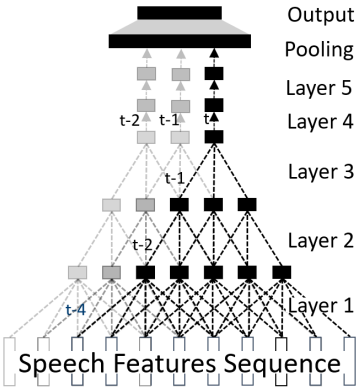

Figure 1: *x-vector speaker embedding. The input sequence of speech features at the top is processed by three TDNN layers. A Statistical Pooling Layer is computed over a speech segment. The Speaker embedding is finally compute by a FFN.*

2.1. Model topology: x-vector

This work is based on the x-vector model structure proposed by Snyder et al. [3], with some simplifications. Compared to the original x-vector model, our architecture, shown on Table 1, uses an increased input feature dimension from 24 to 40, reduces the pooling dimension from 1500 to 512 and removes a fully-connected layer between the embedding and speaker output layer. We reduce the embedding dimension from 512 to 256. In our testing, these modifications did not degrade the recognition performance and result in much lower complexity. We use this topology as a baseline for comparing against the lrx-vector.

Table 1: *Baseline x-vector configuration for a speech utterance with T frames. Three TDNN, two FFN layer are processing the input sequence. The embedding is computes by a FNN on top of the statistical pooling layer.*

	layer context	Affine
Layer1	[t-2,t+2]	$(5 \times 40) \times 512$
Layer2	{t-2,t,t+2}	$(3 \times 512) \times 512$
Layer3	{t-2,t,t+2}	$(3 \times 512) \times 512$
Layer4	{t}	512×512
Layer5	{t}	512×512
Stats pooling	[0,T)	N/A
Segment	[0,T)	1024×256
Output	[0,T)	$256 \times$ N

The output layer is only used during model training; for speaker enrollment and verification, the embedding is taken from the *Segment*-layer (see Table 1). One speaker embedding is computed for an entire utterance, regardless of its length. We use the length-normalized cosine distance of this 256-dimension embedding vectors between enrollment and test utterances to produce the speaker recognition score.

2.2. Loss function

While the conventional softmax loss works reasonably well for training speaker embeddings, it is specifically designed for classification, not verification tasks. Speaker recognition systems trained with softmax loss typically use PLDA [24] in the backend to improve separation between speakers. The triplet loss function, which is designed to reduce intra-speaker and increase inter-speaker distance, has shown to be more effective for speaker recognition [4]. Likewise, the end-to-end loss [5] has better performance than softmax. The downside to these kinds of losses is that the training infrastructure is significantly more complex than one used for supervised learning with softmax. In a prior study [6], we explored the use of several recently proposed loss functions that were first introduced in face recognition research. These loss functions are drop-in replacements for softmax. Thus, modification to training code is simple with little overhead in training speed. We found Additive Margin Softmax (AM-softmax) [25] to perform best in the far-field test set, and incorporating PLDA did not improve performance against the simpler cosine distance. The elimination of the PLDA in the inference pipeline makes the entire model easy to deploy to target hardware, with the help of tools such as the Intel® Distribution of OpenVINO™ toolkit[1].

3. Compact Speaker Embedding System

3.1. Model topology: lrx-vector

First, we analyze the vanilla TDNN layer, which we represent by a feed forward network (FFN). The layer of the baseline x-vector topology described in Section 2.1 processes the input $x_t \in \mathbb{R}^{c \times n}$ at time t. It is a concatenation of c feature vectors according to the layer context. For example, layer 2 of Table 1 is defined by following row:

	layer context	Affine
Layer2	{t-2,t,t+2}	$(3 \times 512) \times 512$

Here, $c = 3$ for layer context $t-2, t, t+2$ of features generated from layer one at 3 time steps. Each input feature to this layer has $n = 512$ dimensions. The overall TDNN output has $m = 512$ dimensions. The input sequence is shifted by one to process the output for the next time step. The FFN representing the TDNN layer with weight matrix $\mathbf{W} \in \mathbb{R}^{(c \times n) \times m}$, but no bias is defined as follows:

$$y = \Phi(\mathbf{W} x_t) \tag{1}$$

where Φ a non linear activation function, i.e., ReLU in this paper. The number of overall weights in a TDNN is $c \cdot n \cdot m$. In the lrx-vector embedding, the TDNN matrix described above is replaced by two matrices $\mathbf{W_a} \in \mathbb{R}^{(c \times n) \times k}$ and $\mathbf{W_b} \in \mathbb{R}^{k \times m}$. The output of the TDNN layer is

$$y = \Phi\left((\mathbf{W_a} \mathbf{W_b}) x_t\right) \tag{2}$$

where W_a and W_b are low-rank representations of W with low-rank constant $1 < k < n$. The overall number of weights in a low-rank TDNN layer is $c \cdot k \cdot (n + m)$ which is significantly smaller compared to vanilla TDNN layer when k is set properly. The lrx-vector configuration is presented in Table 2.

Finally, less weights need to be stored in non volatile memory for lrx-vector system. On the other hand, the compute increases which is most often not an issue on recent DSP platforms including efficient matrix-matrix multiplication units.

3.2. Training

We have explored several different ways of training the lrx-vector system:

1. **Random initialization:** The network is initialized by PyTorch's default random initialization. It is trained in the same way as the baseline x-vector system using the AM-softmax loss described in Section 2.2.

[1] https://docs.openvinotoolkit.org/

Table 2: *lrx-vector system configuration for a T-frame speech utterance. Matrices of the four layers with most weights are replaced by low rank matrices.*

	layer context	Affine
Layer1	[t-2,t+2]	$(5 \times 40) \times 512)$
Layer2	{t-2,t,t+2}	$(3 \times 512) \times k_2, k_2 \times 512$
Layer3	{t-2,t,t+2}	$(3 \times 512) \times k_3, k_3 \times 512$
Layer4	{t}	$512 \times k_4, k_4 \times 512$
Layer5	{t}	$512 \times k_5, k_5 \times 512$
Stats pooling	[0,T)	N/A
Segment	[0,T)	1024×256
Output	[0,T)	$256 \times$ N

2. **lrx-SVD$_0$:** A x-vector baseline system is trained, and singular value decomposition (SVD) is performed on the weight matrices. For each layer, we keep a subset of the singular values. It is similar to work done by Nakkiran et al [12]. The network is not trained any further after SVD.
3. **lrx-SVD$_F$:** A network obtained from SVD$_0$ is fine tuned using the baseline training system until convergence with lowered learning rate.
4. **Knowledge distillation:** This is a method of using a larger teacher network to train a smaller student network to achieve better performance than it is possible with the smaller network alone. It has been successfully applied in computer vision tasks [26]. We find the use of a well-trained full-rank x-vector (i.e. the baseline system described in Section 2) as the teacher to the lrx-vector being particularly effective. Our model training procedure is modified with a loss function combining contributions from knowledge distillation (KD) and AM-softmax (AMS):

$$L_{\text{KD, AMS}} = \alpha L_{\text{KD}} + (1-\alpha) L_{\text{AMS}} \tag{3}$$

Here, L_{KD} can be computed by Kullback Leibler divergence (KLD), Mean Square Error (MSE) or Cosine Similarity (COS). We will make these comparisons in the experiments. Determining the weight α with, e.g., a grid search is time and compute intensive. We partially circumvent this by applying an idea derived from multi-task learning, previously proposed by Du et al. [27]. The KD loss gets minimized as long as its gradient has non-negative cosine similarity with the target gradient. The teacher is ignored, otherwise. Hence, we minimize following equation with Gradient Cosine Similarity (GCS) to train our speaker identification lrx-vector embedding with $\alpha = 0.5$:

$$L_{\text{GCS, KD, AMS}} = \begin{cases} L_{\text{KD, AMS}} & \cos\left(L_{\text{KD}}, L_{\text{AMS}}\right) > 0 \\ L_{\text{AMS}} & \text{otherwise} \end{cases} \tag{4}$$

4. Experiments

Our systems were developed by using Voxceleb 1 and 2. The proposed compact speaker identification system is evaluated using the Voxceleb 1 & 2 data set. The data is described by McLaren et al. [28], Nagrani et al. [29], [30] as training data. Evaluation was performed on the VOiCES challenge far-field text-independent dataset [31]. The VOiCES development set was used to optimize our system. We present results for the development and evaluation sets using equal error rate (EER) and minimum decision cost function (minDCF) metrics as defined in the VOiCES challenge [32].

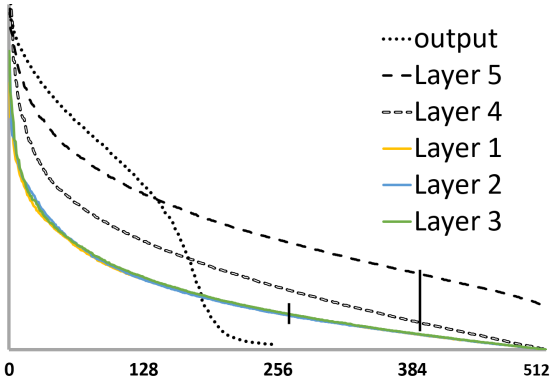

Figure 2: *Singular Values of x-vector model matrices. The low-rank constant is, e.g., $k_2 = k_3 = 256$ and $k_4 = k_5 = 384$.*

Our training data is prepared by applying 4x data augmentation. For each augmented speech file we convolve a randomly chosen room impulse response (RIR) from 100 artificially generated by Pyroomacoustics [33] and 100 selected from Aachen Impulse Response Database [34], then mixing in with randomly chosen clips from Google's Audioset under Creative Commons [35]. The SNR for mixing was uniformly distributed between 0 and 18 dB. We extract 40-dimensional mel-filterbank features from 25 ms frames with 15 ms overlap. The features are mean-normalized on a 3-second sliding window.

Our system was developed using the PyTorch[2] framework. For the baseline system, we used an initial learning rate of 0.1 and decaying to a final learning rate of 0.0001 in 30 epochs of the training data. Training with knowledge distillation and fine-tuning started at lower learning rate of 0.01. A weight decay of $1e-6$ is used in all experiments. We trained all networks until convergence was achieved. This took most often no longer than 20 epochs.

4.1. Low-Rank Factorization

An SVD is used to factorize the matrices of a previously trained x-vector system Figure 2 shows the singular values for each layer. We have empirically figured out that a low rank input layer and low rank layers after the stats pooling significantly decreases the speaker identification accuracy. In contrast, Nakkiran el al. [12] use a low-rank first layer in key-word spotting, successfully. In this paper, we set $k_2 = 0.5n_2$, $k_3 = 0.5n_3$ and $k_3 = 0.75n_3$, $k_4 = 0.75n_3$ where n_2, n_3, n_4 and n_5 are the dimension of the input vectors to layer 2 to 4. It cannot be ruled out that a full grid-search finds better choices for k_i given a target number of overall weights in the lrx-vector. Automatically determining optimal k_i is subject to our further research.

Table 3 compares scaled full-rank x-vector to the randomly initialized lrx-vector, lrx-SVD$_0$ and lrx-SVD$_F$. Note that we linearly scaled the output dimension of each layer in the x-vector for all layer with the same constant factor before the stats pooling, in order to obtain the same model size of $550k$ parameters as in the lrx-vector. The results indicate that SVD, even with fine-tuning, did not produce better results than random initialization. Next, we focus the attention to another technique.

[2]https://pytorch.org/

Table 3: *Factorization with Singular Value Decomposition (SVD) compared to training from scratch*

550k	Dev		Eval	
weights	EER	minDCF	EER	minDCF
full rank	2.78	0.307	**6.69**	**0.483**
lrx-Random init	**2.73**	**0.289**	6.76	0.484
lrx-SVD_0	3.23	0.335	7.39	0.539
lrx-SVD_F	2.74	**0.289**	6.76	0.484

4.2. Training with Knowledge Distillation

For our research, we choose the teacher to be the baseline system as described in Section 2. The EER and minDCF of the teacher system in Table 4 is a strong baseline, competitive with the top x-vector systems of comparable complexity in the VOiCES Challenge [32]. We use this baseline system to teach a student lrx-vector system with $550k$ weights, as described in the previous section.

In the first set of results on Table 4, we present the experiments using the standard KD learning objective from Eq. 3, with $\alpha = 0.5$. It can be seen that KD is generally an improvement over random initialization or lrx-SVD_F, with KD-MSE performing best with relative EER improvement of 7.19% on the development, and 5.68% on the evaluation sets. Although this set of results is promising, it might be possible to improve further by tuning α. Practically, doing this by grid search will require long training time.

Table 4: *Comparison of loss functions between teacher and student at the example of the lrx-vector system with overall 550k weights. The teacher is a baseline system with 4800k weights.*

lrx-vector	Dev		Eval	
550k weights	EER	minDCF	EER	minDCF
lrx-SVD_F	2.74	0.289	6.76	0.484
KD-KLD	2.62	0.290	6.72	0.484
KD-MSE	**2.58**	**0.270**	**6.31**	**0.452**
KD-COS	2.67	0.273	6.37	0.456
Teacher	1.83	0.189	5.5	0.381

Table 5 shows the results of gradient cosine similarity as described in Section 3.2. Here, we see that the best performance is achieved with cosine similarity distillation loss, with relative EER improvement of 11.15% for development and 3.74% for evaluation sets over lrx-SVD_F.

Table 5: *Knowledge distillation with positive Gradient Cosine Similarity between teacher and student. Otherwise, only Additive Margin Softmax speaker identification loss.*

lrx-vector	Dev		Eval	
550k weights	EER	MinDCF	EER	MinDCF
lrx-SVD_F	2.74	0.289	6.76	0.484
KD-GCS-KLD	2.45	0.270	6.77	0.487
KD-GCS-MSE	**2.42**	0.272	6.49	0.470
KD-GCS-COS	2.47	**0.266**	**6.44**	**0.458**
Teacher	1.83	0.189	5.5	0.381

4.3. Scaling lrx-vector

Comparing x- and lrx-vector based speaker identification of different sizes at approximately the same EER is subject of this evaluation section. The number of weights in the models were adjusted by changing the dimension of each hidden layers by a multiplication factor < 1. This factor is the same for all layers in the network up to the stats pooling layer. Automatically determining Layer dependent factors is subject of future research. lrx-vector as well as x-vector systems were trained by knowledge distillation using KD-GCS-COS. We selected the best models given the development set as similar in previous sections. The results are shown in Table 6. A 2.1% EER was achieved with an lrx-Vector system that requires to store $800k$ weights in ROM. This is 73% of the size to a comparable x-vector system that meets the same EER.

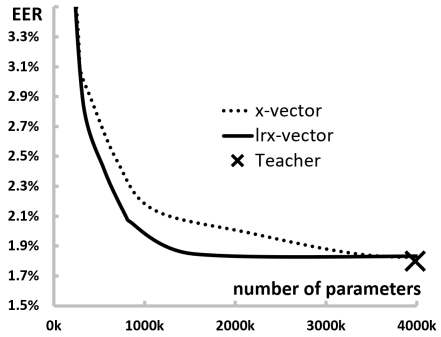

Figure 3: *Achieved Equal Error Rate (EER) on the development set at different number of weights of X- and lrx-Vector Systems.*

Table 6: *Weigh reduction of lrx-vector system compared to x-vector system that achieves same EER.*

EER	lrx-vector size	
Dev	% of x-vector equivalent	#weights
4.5	100	134k
2.4	79	550k
2.1	73	800k
1.8	72	1540k
1.83	X-Vector Teacher with 4800k weights	

Figure 3 illustrates the memory saving of lrx-vector systems compared to x-vector systems at different number of weights. As expected, there is no improvement over the teacher system when the model is very large, i.e., when the model capacity hits the upper bound of the task. Very small models do not benefit from low rank matrices, too. What can be seen is an improvement of the lrx-vector over the x-vector system at common operation points of the system. In other words, our proposed lrx-vector system shifts the operation point towards smaller models.

5. Conclusion

This paper addresses compact speaker identification by lrx-vector embedding. We propose a low-rank version of the popular TDNN based x-vector embedding where big matrices are replaced by low-rank matrices. We address one of the main bottlenecks of low power inference in small edge devices, memory access, by reducing the size of the model. Using the VOiCES far-field test set, we achieved 28% reduction in the number of parameters compared to the full size model, at the same EER of 1.8%. The lrx-vector is also shown to achieve reduction in model size compared to scaled-down x-vector, at comparable EERs across a wide range of operating points. Future research beyond this work can include other means of searching for best knowledge distillation hyper-parameter α, and joint low-rank and weight quantization optimizations.

6. References

[1] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5796–5800.

[2] A. Zhang, Q. Wang, Z. Zhu, J. Paisley, and C. Wang, "Fully supervised speaker diarization," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6301–6305.

[3] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[4] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep speaker: an end-to-end neural speaker embedding system," *arXiv preprint arXiv:1705.02304*, 2017.

[5] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[6] J. Huang and T. Bocklet, "Intel Far-Field Speaker Recognition System for VOiCES Challenge 2019," in *Proc. Interspeech 2019*, 2019, pp. 2473–2477.

[7] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. CONF. IEEE Signal Processing Society, 2011.

[8] M. Georges, S. Kanthak, and D. Klakow, "Accurate client-server based speech recognition keeping personal data on the client," in *ICASSP*, 2014.

[9] G. Stemmer, M. Georges, J. Hofer, P. Rozen, J. G. Bauer, J. Nowicki, T. Bocklet, H. R. Colett, O. Falik, M. Deisher, and S. J. Downing, "Speech recognition and understanding on hardware-accelerated DSP," in *Interspeech 2017*, 2017, pp. 2036–2037.

[10] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," in *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds. Curran Associates, Inc., 2015, pp. 442–450.

[11] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition," *CoRR*, vol. abs/1402.1128, 2014.

[12] P. Nakkiran, R. Alvarez, R. Prabhavalkar, and C. Parada, "Compressing deep neural networks using a rank-constrained topology," in *Proceedings of Annual Conference of the International Speech Communication Association (Interspeech)*, 2015, pp. 1473–1477.

[13] W. Chen, J. Wilson, S. Tyree, K. Weinberger, and Y. Chen, "Compressing neural networks with the hashing trick," in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, F. Bach and D. Blei, Eds., vol. 37. Lille, France: PMLR, 07–09 Jul 2015, pp. 2285–2294.

[14] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[15] Jingyang Zhu, Zhiliang Qian, and Chi-Ying Tsui, "Lradnn: High-throughput and energy-efficient deep neural network accelerator using low rank approximation," in *2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC)*, Jan 2016, pp. 581–586.

[16] R. Sahraeian and D. Van Compernolle, "A study of rank-constrained multilingual dnns for low-resource asr," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 5420–5424.

[17] P. Dighe, A. Asaei, and H. Bourlard, "Low-rank and sparse soft targets to learn better dnn acoustic models," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2017, pp. 5265–5269.

[18] S. Chandra, N. Usunier, and I. Kokkinos, "Dense and low-rank gaussian crfs using deep embeddings," in *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[19] Z. Ding and Y. Fu, "Deep domain generalization with structured low-rank constraint," *IEEE Transactions on Image Processing*, vol. 27, no. 1, pp. 304–313, Jan 2018.

[20] A. V. Miceli Barone, "Low-rank passthrough neural networks," *Proceedings of the Workshop on Deep Learning Approaches for Low-Resource NLP*, 2018.

[21] V. Sharan, K. S. Tai, P. Bailis, and G. Valiant, "Compressed factorization: Fast and accurate low-rank factorization of compressively-sensed data," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019, pp. 5690–5700.

[22] J. Zhang, J. Huang, M. Deisher, H. Li, and Y. Chen, "Structural sparsification for far-field speaker recognition with gna," 2019.

[23] M. Georges, K. Czarnowski, and T. Bocklet, "Ultra-Compact NLU: Neuronal Network Binarization as Regularization," in *Proc. Interspeech 2019*, 2019, pp. 809–813.

[24] S. Ioffe, "Probabilistic linear discriminant analysis," in *Computer Vision – ECCV 2006*, A. Leonardis, H. Bischof, and A. Pinz, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 531–542.

[25] F. Wang, J. Cheng, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[26] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015. [Online]. Available: http://arxiv.org/abs/1503.02531

[27] Y. Du, W. M. Czarnecki, S. M. Jayakumar, R. Pascanu, and B. Lakshminarayanan, "Adapting auxiliary losses using gradient similarity," 2018.

[28] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The 2016 speakers in the wild speaker recognition evaluation," in *Interspeech 2016*, 2016, pp. 823–827.

[29] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Computer Science and Language*, 2019.

[30] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *INTERSPEECH*, 2017.

[31] C. Richey, M. A. Barrios, Z. Armstrong, C. Bartels, H. Franco, M. Graciarena, A. Lawson, M. K. Nandwana, A. Stauffer, J. van Hout, and et al., "Voices obscured in complex environmental settings (voices) corpus," *Interspeech 2018*, Sep 2018. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1454

[32] M. K. Nandwana, J. Van Hout, M. McLaren, C. Richey, A. Lawson, and M. A. Barrios, "The voices from a distance challenge 2019 evaluation plan," *arXiv preprint arXiv:1902.10828*, 2019.

[33] R. Scheibler, E. Bezzam, and I. Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 351–355.

[34] M. Jeub, M. Schafer, and P. Vary, "A binaural room impulse response database for the evaluation of dereverberation algorithms," in *2009 16th International Conference on Digital Signal Processing*. IEEE, 2009, pp. 1–5.

[35] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 776–780.