INCREASING COMPACTNESS OF DEEP LEARNING BASED SPEECH ENHANCEMENT MODELS WITH PARAMETER PRUNING AND QUANTIZATION TECHNIQUES

Jyun-Yi Wu[*], *Cheng Yu*[*], *Szu-Wei Fu*[†], *Chih-Ting Liu*[*], *Shao-Yi Chien*[*], *Yu Tsao*[†]

[*]GIEE, National Taiwan University, Taiwan
[†]CITI, Academia Sinica, Taiwan

ABSTRACT

The most recent studies on deep learning based speech enhancement (SE) are focused on improving denoising performance. However, successful SE applications require require striking a desirable balance between the denoising performance and computational cost in real scenarios. In this study, we propose a novel parameter pruning (PP) technique, which removes redundant channels in a neural network. In addition, parameter quantization (PQ) and feature-map quantization (FQ) techniques were also integrated to generate even more compact SE models. The experimental results show that the integration of PP, PQ, and FQ can produce a compacted SE model with a size of only 9.76 % compared to that of the original model, resulting in minor performance losses of 0.01 (from 0.85 to 0.84) and 0.03 (from 2.55 to 2.52) for STOI and PESQ scores, respectively. These promising results confirm that the PP, PQ, and FQ techniques can be used to effectively reduce the storage of an SE system on edge devices.

Index Terms— Compactness, Parameter Pruning, Parameter Quantization, Low Computational Cost

1. INTRODUCTION

The goal of speech enhancement (SE) is to generate enhanced speech with better quality and intelligibility over the original noisy speech with many SE methods having been proposed in the past. Traditional SE approaches are derived from the characteristics of speech and noise signals; well-known examples include the spectral subtraction [1], Wiener filter [2], short-time spectral amplitude estimators [3], and maximum-likelihood spectral amplitude [4] algorithms. These traditional approaches perform well when the assumed properties of speech and noise signals are maintained. However, their performance degrades notably when dealing with non-stationary noises or in very low signal-to-noise ratio (SNR) conditions.

Recently, deep learning algorithms have been successfully introduced to the SE field [5]. Generally, a deep-learning model is used as a mapping function with the aim of transforming noisy speech into clean speech. Notable approaches include the deep denoising auto-encoder (DDAE) [6], deep fully-connected neural network [7], convolutional neural network (CNN) [8], and long short-term memory model (LSTM) [9]; all of these models have shown promising results for transforming noisy spectral features into clean ones. More recently, several studies proposed the use of convolutional structures for speech and audio signal analysis and reconstruction [10, 11, 12, 13, 14], and thus the SE tasks can be carried out in the time domain. One well-known model is the fully convolutional neural network (FCN), which consists of multiple filters, each formed by plural channels [10, 11].

Numerous studies have confirmed the outstanding denoising capability of deep learning-based methods, especially under more challenging conditions (e.g., non-stationary noises and low SNR conditions). However, a notable disadvantage of deep learning-based solutions is the requirement of a large storage space for the SE models, which makes them difficult to implement for devices with limited resources. In [25], an effective teacher-student architecture is proposed where the outputs from a large, high-performance deep neural network (DNN) are used to guide the training of a smaller DNN model in a sequential or multi-task transfer learning manner. This paper, however, proposes two techniques, namely parameter pruning (PP) and parameter quantization (PQ), that aim to directly increase the compactness of a well-trained deep learning-based SE models. The PP technique removes redundant channels, while the PQ technique groups and represents similar weights using a cluster centroid. To evaluate the effectiveness of these two techniques, we used the TIMIT database [15] with several noise sources.The experimental results show that both PP and PQ techniques can effectively improve the model compactness with modest degradations in quality and intelligibility. intelligibility.

2. RELATED RESEARCH

Many algorithms have been derived to increase the compactness of neural network models, such as pruning, sparse constraints, and quantization. Pruning algorithms are designed to reduce the network complexity and address overfitting [16] by reducing redundant components. The approaches that utilize sparse constraints build compact models by reducing trivial

Thanks to IOX center, NTU.

filters in the original deep-learning models [17]. Quantization algorithms, on the other hand, compress the size of the original network by reducing the number of bits required to represent each weight [18, 19]. These aforementioned model compression methods significantly reduced memory usage with only a modest loss in recognition accuracy.

Based on our literature survey, only a few studies have investigated potential approaches to increase the compactness of SE models. Sun and Li proposed the use of a quantization technique to increase the compactness of an SE model [20]. Ko et al. investigated the correlation of precision scaling and neuron numbers in an SE model [21]. In [22], a two-stage quantization approach was derived to optimally reduce the bit number when the parameters are encoded in floating point representation. In the present study, the PP technique adopts a different and novel concept that directly removes redundant channels to form a compact FCN model. The size of this model is then reduced further with the PQ technique.

3. THE PROPOSED PP AND PQ TECHNIQUE

This section introduces the proposed PP and PQ techniques, as well as their integration.

3.1. Parameter Pruning (PP) Technique

3.1.1. FCN-based Waveform Mapping

Figure 1(a) shows the process of the waveform mapping based on the FCN model. In the figure, we have J filters: $\{\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_J\}$; $\mathbf{F}_j \in \mathbb{R}^{L\times I}$ is the j-th filter, and $\mathbf{F}_{ji} \in \mathbb{R}^{L\times 1}$ is the i-th channel of $\mathbf{F}_j$. $\mathbf{F}_{ji} = (w_1, w_2, ..., w_L)_{L\times 1}$ where w_l is the channel weight. It is assumed that the receptive field and output sample of filter $\mathbf{F}_j$ is $\mathbf{R}(t) \in \mathbb{R}^{L\times I}$ and $\mathbf{y}_j(t)$, respectively. The resulting convolution operation is:

$$\mathbf{y}_j(t) = \sum_{i=1}^{I} \mathbf{F}_{ji}^T \mathbf{R}_i(t). \quad (1)$$

3.1.2. Definition of Sparsity

We estimate the redundancy based on the sparsity [17] of each channel in a filter. For filter $\mathbf{F}_j$ in an arbitrary layer, we first compute the mean absolute value of all filter weights:

$$M_{F_j} = \frac{\sum_I(\sum_L |w|)}{I \times L}, \quad (2)$$

where I and L are the channel number in a filter and weight number in a channel, respectively; w is a weight parameter. The sparsity of the i-th channel of $\mathbf{F}_j$ can then be defined as:

$$S(i) = \frac{\sum_{l=1}^{L} \sigma(w_l)}{L}, \quad (3)$$

$$\sigma(X) = \begin{cases} 1, & \text{if } |X| < M_{Fj} \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

When $S(i)$ is close to 1, most of weights in a channel are smaller than M_{F_j}, and thus the channel is considered redundant.

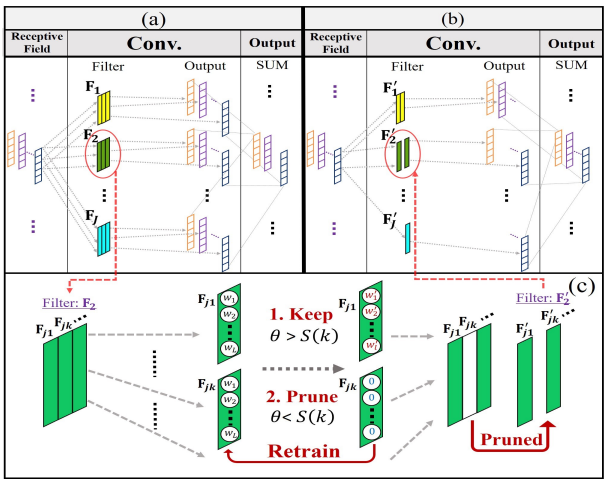


Fig. 1. The PP process: (a) original model; (b) pruned model, and (c) the pruning and retraining process.

3.1.3. Channel Pruning

In the PP technique, the pruning mechanism contains a retraining step. As shown in Fig.1.(c), if the sparsity $S(k)$ in some channels $\mathbf{F}_{jk}$ are larger than a predefined threshold θ, the weights within the channel $\mathbf{F}_{jk}$ are set to zero. Next, we retrain the model and remove $\mathbf{F}_{jk}$ after several iterations. As shown in Fig.1.(b), we can obtain $\mathbf{F}'$ as the channel-pruned filters. Because $\mathbf{F}'_{ji}$ is reduced, $\mathbf{R}'_i$(t) is reduced accordingly. Finally, the output can be computed as follows:

$$\mathbf{y}'_j(t) = \sum_{i=1}^{I-K} \mathbf{F}'^T_{ji} \mathbf{R}'_i(t), \quad (5)$$

where K is the number of pruned channels. This PP technique ensures that the compacted model remains stable after the pruning steps, while the retraining steps make models adjustable to the zero-weighted channels. We believe that this approach, unlike other pruning methods that directly remove filters, can effectively prevent severe performance drops.

3.2. Parameter Quantization (PQ) Technique

In this study, the PQ is carried out based on the k-means algorithm. By applying the k-means algorithm, the parameters in a neural network model are grouped into several clusters, where each cluster of parameters shares a centroid value. Fig. 2 shows an example of the k-means-based PQ process. In this figure, each weight parameter in the original model is represented by a 32-bits floating point number. By applying the k-means with k=4, we can obtain a look-up table with 4 cluster centroids. Each weight in the model is then denoted with a cluster index that is linked to the corresponding cluster centroid. Therefore, the 10 weights (each represented as a 32-bit floating point number) in the original model can be represented with 4 cluster indices and 4 centroids. The corresponding compression rate is: $(10 * 32)/(4*32 + 2*10) = 2.16$.

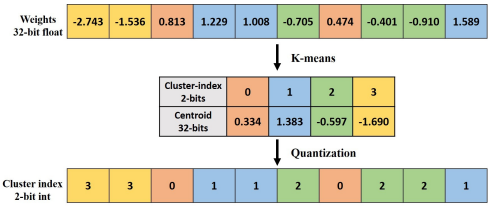


Fig. 2. An example of the PQ technique.

3.3. Integration of PP and PQ

Although both methods aim to increase the model compactness, the PP and PQ techniques are derived based on different concepts. The compatibility of these two techniques were analyzed accordingly. The PP is applied to remove redundant channels and establish a compact SE model, and the PQ is subsequently used to further quantize the model parameters. Note that the proposed PP and PQ can be combined with other existing model compression techniques. In this study, we tested the compatibility of the PP and PQ with a feature-map quantization (termed FQ) technique. More specifically, the FCN model is first processed by PP and PQ; then, the input values and feature maps are further quantized to lower precision. The results of the integration of PP, PQ, and FQ are reported in Section 4.

4. EXPERIMENTS

In this section, we first introduce the experimental setup and exhibit the experimental results.

4.1. Experimental Setup

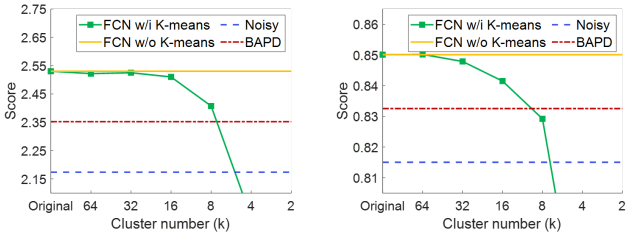
The TIMIT corpus was used to prepare the training and test sets. All 4620 utterances in the TIMIT training set were selected as training data. These utterances were corrupted with five noise types (Babble, Car, Jackhammer, Pink, and Street) at five SNR levels (-10 dB, -5 dB, 0 dB, 5 dB, and 10 dB). 100 utterances were then randomly selected from the TIMIT testing set to be used as the testing data. These utterances were artificially corrupted with three additional noise types (Babycry, White, and Engine) at four SNR levels (-12 dB, -6 dB, 0 dB, and 6 dB). Note that we intentionally designed mismatching noise types and SNR levels for training and testing conditions in order to simulate a more realistic scenario. All of the speech and noise signals were recorded in a 16 kHz/16 bit format. We evaluated the PP and PQ techniques with the following standardized metrics: perceptual evaluation of speech quality (PESQ) [23] and short-time objective intelligibility (STOI) [24]. PESQ measures the quality of the processed speech by assigning a score ranging from 0.5 to 4.5; a higher PESQ score denotes better speech quality. STOI measures speech intelligibility by assigning a score ranging from 0 to 1; a higher STOI score denotes better intelligibility.

We followed the setup in [11] to build the FCN-based SE system. The FCN model had 7 hidden layers; each layer containing 30 1-D convolutional filters (with the size of 55). The last layer contained only 1 filter. Batch normalization with mode-2 and the Leaky-ReLU activation function were used. The parameters were trained using the Adam optimizer. Without compression, the FCN model obtained 2.55 and 0.85 PESQ and STOI scores, respectively. For comparisons purposes, a fully connected network SE system was built, whose model architecture and training setup were the same as that used in [7]. The model had three dense layers of 2048 neurons and used the ReLU activation function. The dropout process was applied, with a rate of 0.2. One dense layer, with a linear activation function, was used to generate the enhanced speech. The Adam optimizer was used to estimate the model parameters. When using such a fully connected network, the PESQ and STOI scores were 2.32 and 0.83, respectively.

4.2. Experimental Results

4.2.1. Parameter Quantization (PQ)

For the PQ technique, the number of clusters k was set to 2, 4, 8, 16, 32, and 64, and the corresponding PESQ and STOI results are shown in Fig. 3 (a) and (b), respectively. It is clear that the scores of both PESQ and STOI decrease when the cluster number k is reduced. In practical SE applications, performance and computational cost must be considered simultaneously. Thus, we may first define a bound for acceptable performance drop (BAPD) and continue reducing the number of clusters until the evaluation scores fall below the defined bound. In this experiment, we consider this BAPD to be the average score of the results produced with the original SE model and that of noisy speech. Using Fig.3(a) as an example, the PESQ scores for noisy speech and FCN without pruning were 2.15 and 2.55, respectively. The BAPD was then defined as (2.15+2.55)/2 = 2.35. Figs. 3(a) and (b) show that the PESQ and STOI scores are similar with BAPD when k>4. Note that the BAPD value used in the present study is just an example. Users can arbitrarily specify the BAPD value to meet the hardware requirements.

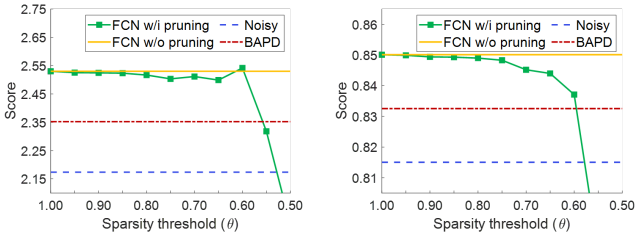


(a) PESQ (b) STOI

Fig. 3. The average PESQ and STOI scores yielded from the PQ technique with different numbers of clusters. BAPD denotes the bound for acceptable performance drop.

4.2.2. *Parameter Pruning (PP)*

When implementing the PP technique, the sparsity threshold gradually reduced from 1 (i.e., without conducting PP) to 0.60 with a step size of 0.05. The model was retrained after each sparsity threshold reduction. The PESQ and STOI results are shown in Fig. 4 (a) and (b), respectively. The resulting STOI scores show a clear drop when the sparsity threshold was decreased from 0.65 to 0.60, while PESQ scores were slightly increased. In Table 1, we listed the correlation between the sparsity threshold and the removal ratio in the SE model. The results in Table 1 show that the corresponding removal ratio is 21.8% when the sparsity threshold is set to 0.65.



(a) PESQ (b) STOI

Fig. 4. The average PESQ and STOI scores yield by the PP technique with different sparsity threshold values. BAPD denotes the bound for acceptable performance drop.

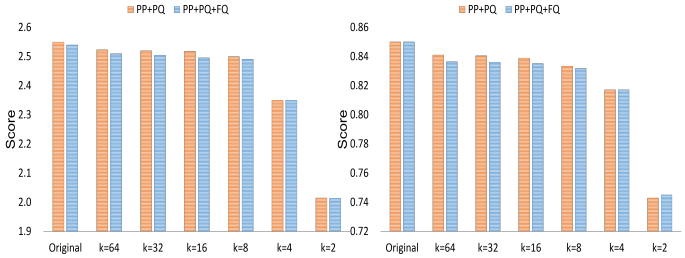
Table 1. Correlation of sparsity threshold and removal ratio, and the number of remaining parameters in the SE model.

Sparsity threshold	Removal ratio	Remaining parameters
1.00	0.00%	300,300
0.70	14.0%	258,170
0.65	21.8%	234,850
0.60	26.4%	221,155
0.55	36.8%	189,640

4.2.3. *The Integration of PP and PQ*

Whether the integration of the PP and PQ techniques can provide an even more compact SE model was also investigated. Based on our preliminary experiments, a more effective order of integration is to use PP followed by PQ. From the results in Fig. 4, setting the sparsity threshold as θ=0.65 can still return reasonably satisfactory performance. Therefore, we tested the integration of the PP technique with θ=0.65 and the PQ technique by varying the number of clusters. The results are shown in Fig. 5 with the notation of "PP+PQ". From the figure, we note that the systems with θ=0.65 suffered considerable performance drops when k=8, while the system with θ=0.65 and k=16 achieved the optimal balance between denoising performance and computational cost: The size of the compacted SE model was only 9.76 % as compared with that of the original model, where the STOI score was reduced by 0.01 (from 0.85 to 0.84) and the PESQ score was reduced by 0.03 (from 2.55 to 2.52).

In addition to PP and PQ, FQ was also applied, which quantizes the original 32 bits into 16 bits. The results of using FQ are also listed in Fig. 5 with the notation of "PP+PQ+FQ". From the figure, we note that with FQ, there is no clear degradation (the orange bars of "PP+PQ" and blue bars of "PP+PQ+FQ" are almost the same), confirming the compatibility of PP and PQ with FQ. Note that by applying FQ, the bit number of centroids for the PQ is reduced to 50%.[1] We also tested FQ to quantize the feature-map to 8 bits, but the performance degraded notably. Owing to the space limitation, we did not report the 8-bit results in Fig. 5.



(a) PESQ (b) STOI

Fig. 5. The average PESQ and STOI scores achieved by PP+PQ (orange bar) and PP+PQ+FQ (blue bar) .

5. CONCLUSION

We propose utilizing the PP and PQ techniques to increase the compactness of the FCN model for performing time-domain SE. The main contribution of this study is two-fold. First, to the best of our knowledge, the PP technique is the first technique that directly removes redundant channels in the FCN model. Second, we have shown that by applying PP, PQ, and an integration of PP and PQ effectively reduces the model size with only a modest performance drop. The results suggest that PP and PQ techniques can facilitate an SE system with a compact structures to be installed in edge devices that have lower storage. Note that although compression techniques for deep-learning models applied to pattern recognition (classification) tasks have been extensively studied, only few works have addressed model compression for signal regression tasks. Because of their different output formats, the effects of model compression on regression tasks are very different from that on classification tasks. This study is a first for investigating the effects of model pruning/quantization on the SE task (a regression task), and the results can be used as a useful guidance for future SE studies. Moreover, we have confirmed the compatibility of the PP and PQ with FQ, another model quantization technique that applies quantization on inputs and feature-map parameters. In this study, attention was focused toward handling additive noises. Future work will aim to explore the applications of the proposed pruning and quantization techniques on deep learning models for more complicated tasks where additive noise, reverberation and distant talking effects, are all present simultaneously.

[1]The codes, trained FCN models, and dataset used in this study are available via: https://github.com/WilliamYu1993/ICSE

6. REFERENCES

[1] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 113–120, 1979.

[2] Wiener, Norbert, "Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications," *MIT Press*, 1950.

[3] Y. Ephraim and D. Malah, "Speech enhancement using a minimum-mean square error short-time spectral amplitude estimator," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 32, no. 6, pp. 1109–1121, 1984.

[4] R. McAulay and M. Malpass, "Speech enhancement using a soft-decision noise suppression filter," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, no. 2, pp. 137–145, 1980.

[5] D. Wang and J. Chen, "Supervised speech separation based on deep learning: An overview," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 10, pp. 1702–1726, 2018.

[6] X. Lu, Y. Tsao, S. Matsuda, and C. Hori, "Speech enhancement based on deep denoising autoencoder," in *Proc. INTERSPEECH*, 2013.

[7] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "A regression approach to speech enhancement based on deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 1, pp. 7–19, 2015, `https://github.com/yongxuUSTC/sednn/tree/master/mixture2clean_dnn`.

[8] S.-W. Fu, Y. Tsao, and X. Lu, "SNR aware convolutional neural network modeling for speech enhancement," in *Proc. INTERSPEECH*, 2016.

[9] Z. Chen, S. Watanabe, H. Erdogan, and John R. Hershey, "Speech enhancement and recognition using multi-task learning of long short-term memory recurrent neural networks," *Nuclear Physics A*, vol. 2015-January, pp. 3274–3278, 2015.

[10] S.-W. Fu, Y. Tsao, X. Lu, and H. Kawai, "Raw waveform-based speech enhancement by fully convolutional networks," in *Proc. APSIPA*, 2017.

[11] S.-W. Fu, T. Wang, Y. Tsao, X. Lu, and H. Kawai, "End-to-end waveform utterance enhancement for direct evaluation metrics optimization by fully convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 9, pp. 1570–1584, 2018, `https://github.com/JasonSWFu/End-to-end-waveform-utterance-enhancement`.

[12] Ashutosh Pandey and D. Wang, "A new framework for supervised speech enhancement in the time domain," in *Proc. INTERSPEECH*, 2018.

[13] Yi Luo and Nima Mesgarani, "Tasnet: Surpassing ideal time-frequency masking for speech separation," in *Proc. ICASSP*, 2018.

[14] Emad M. Grais, D. Ward, and Mark D. Plumbley, "Raw multi-channel audio source separation using multi-resolution convolutional auto-encoders," in *Proc. EUSIPCO*, 2018.

[15] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. Y. Dahlgren, "Darpa timit acoustic-phonetic continous speech corpus cd-rom. nist speech disc 1-1.1," *NASA STI/Recon Technical Report N*, vol. 93, pp. 27403, 1993.

[16] S. Han, J. Pool, J. Tran, and W. J. Dally, "Learning both weights and connections for efficient neural networks," in *Proc. NIPS*, 2015.

[17] C.-T. Liu, Y.-H. Wu, Y.-S. Lin, and S.-Y. Chien, "Computation-performance optimization of convolutional neural networks with redundant kernel removal," in *Proc. ISCAS*, 2018.

[18] D. Miyashita, E. H. Lee, and B. Murmann, "Convolutional neural networks using logarithmic data representation," *arXiv preprint arXiv:1603.01025*, 2018.

[19] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding," *CoRR*, vol. abs/1510.00149, 2015.

[20] H. Sun and S. Li, "An optimization method for speech enhancement based on deep neural network," in *Proc. IOP Conference Series: Earth and Environmental Science*. IOP Publishing, 2017, vol. 69, p. 012139.

[21] J. H. Ko, J. Fromm, M. Philipose, I. Tashev, and S. Zarar, "Precision scaling of neural networks for efficient audio processing," *CoRR*, vol. abs/1712.01340, 2017.

[22] Y.-T. Hsu, Y.-C. Lin, S.-W. Fu, Y. Tsao, and T.-W. Kuo, "A study on speech enhancement using exponent-only floating point quantized neural network (eofp-qnn)," in *Proc. SLT*, 2018.

[23] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (pesq)-a new method for speech quality assessment of telephone networks and codecs," in *Proc. ICASSP*, 2001.

[24] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of timefrequency weighted noisy speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.

[25] S.-C. Wang, K. Li, Z. Huang, S.M. Siniscalchi, and C.-H. Lee, "A transfer learning and progressive stacking approach to reducing deep model sizes with an application to speech enhancement," in *Proc. ICASSP*, 2017.