



Model Compression by Iterative Pruning with Knowledge Distillation and Its Application to Speech Enhancement

Zeyuan Wei^{1*}, Hao Li^{2*}, Xueliang Zhang¹

¹Department of Computer Science, Inner Mongolia University ²Department of Electrical and Electronic Engineering, Southern University of Science and Technology
weizeyuan@mail.imu.edu.cn, lih9@sustech.edu.cn, cszxl@imu.edu.cn

Abstract

Over the past decade, deep learning has demonstrated its effectiveness and keeps setting new records in a wide variety of tasks. However, good model performance usually leads to a huge amount of parameters and extremely high computational complexity which greatly limit the use cases of deep learning models, particularly in embedded systems. Therefore, model compression is getting more and more attention. In this paper, we propose a compression strategy based on iterative pruning and knowledge distillation. Specifically, in each iteration, we first utilize a pruning criterion to drop the weights which have less impact on performance. Then, the model before pruning is used as a teacher to fine-tune the student which is the model after pruning. After several iterations, we get the final compressed model. The proposed method is verified on gated convolutional recurrent network (GCRN) and long short-term memory (LSTM) for single channel speech enhancement tasks. Experimental results show that the proposed compression strategy can dramatically reduce the model size by 40x without significant performance degradation for GCRN.

Index Terms: pruning, quantization, knowledge distillation, speech enhancement

1. Introduction

In the real environment, noise is inevitable and affects many applications, such as mobile communication and automatic speech recognition. Speech enhancement aims to reduce noise and improve speech quality and intelligibility [1]. Compared with the conventional algorithms [2], deep learning-based speech enhancement developed very fast and achieved great success [3]. However, the large model requires huge computing and memory resources, which restrict its real applications to embedded systems in industry, such as mobile phones and earbuds. Meanwhile, direct training of a small model has unsatisfactory performance.

In order to expand the application scenarios of deep learning, model compression is getting more and more attention recently [4, 5]. Model compression, as its name indicates, is to compress a large model to a small one which has similar performances. In general, compression techniques include *quantization*, *low-rank approximation*, *network pruning* and *knowledge distillation*. The quantization is to convert floating-point numbers into fixed-point numbers with fewer bits. The commonly used quantitative methods include linear, aware quantization proposed in [6] and k-means clustering quantization [7, 8]. Low-rank approximation [9] is to decompose a large parameter matrix into several small ones by using classic math, such as singular value decomposition [10]. Network pruning

was proposed many years ago [11] and widely studied recently. It is based on the observation that many parameters in a large neural network are unimportant. Pruning method is to find those parameters through certain rules to increase the sparsity of the network. Common pruning strategies include one-shot pruning [12] and iterative pruning [13]. In [14], iterative pruning is used to compress speech enhancement models, and trainable parameters of the models were reduced by more than 70% without significantly reducing the enhancement performance. Depending on the granularity, neurons or connections, there are structured [15] and unstructured pruning [16]. Different from the above three compression methods, knowledge distillation [17, 18, 19] is a learning framework that utilizes a teacher-student structure. Generally speaking, the teacher network is an efficient neural network or a collection of neural networks. The student network is a compact neural network. The output of teacher network is used as label to train student network. In [20, 21], knowledge distillation is applied to speech enhancement models. But the purpose is not for model compression.

In this paper, we combine the characteristics of iterative pruning, quantization and knowledge distillation to compress speech enhancement networks. We believe that the model before each pruning is also useful for the restoration of the model after. Therefore, we use the model before pruning as the teacher model to fine-tune the student model after pruning, by combining the outputs of the teacher model with the true labels. After that, clustering-based quantization is employed as well to further reduce the size of the neural network.

In order to evaluate the proposed method, we choose two representative models, LSTM and Gated CRN [22], for speech enhancement. Experimental results show that the model size is greatly reduced about 10x and 40x for LSTM and GCRN respectively without significant performance loss.

The paper is organized as following. In section 2, we briefly introduce the deep learning-based speech enhancement and describe the proposed method of model compression. Experimental setup and results are given in section 3. We conclude the whole work in section 4.

2. Method

2.1. DNN for speech enhancement

Given a clean speech s and a noise n , the noisy speech can be modeled as:

$$m = s + n \quad (1)$$

where $\{m, s, n\} \in \mathbb{R}^{T \times 1}$, and T represents time samples, respectively. The purpose of single channel speech enhancement is to obtain estimated clean speech \hat{s} , with a mapping function:

$$\hat{s} = f_{\theta}(m) \quad (2)$$

* The first two authors have equal contributions to this work

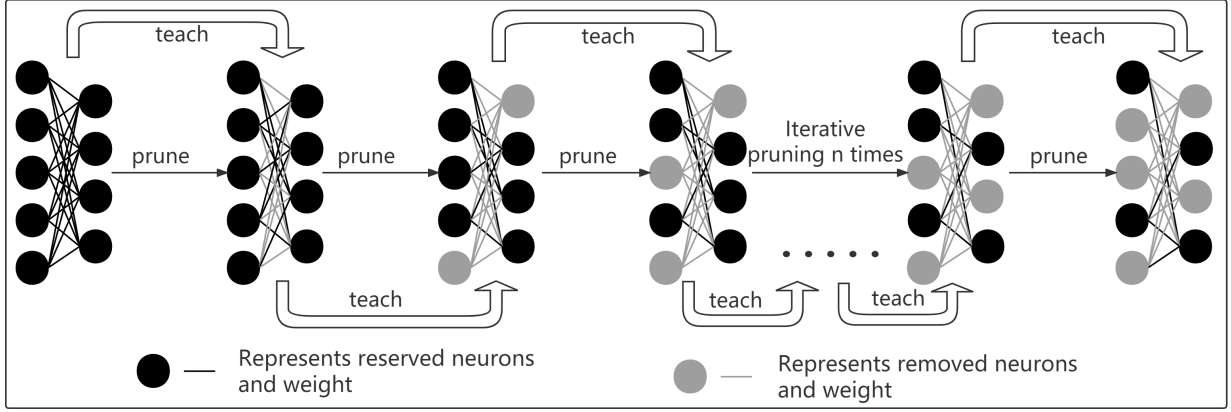


Figure 1: Architectures of iterative knowledge distillation.

where f_θ is the mapping function, indicates GCRN/LSTM in this paper. θ indicates model weights, respectively.

Vanilla GCRN [22] can achieve a satisfactory result. However, the large size of the model and the amount of computation limit the model deployment in an embedded system. The LSTM-based model used in this paper consists of 2 unidirectional LSTM layers with 256 units each and 2 fully connected (FC) layers with 128 units each, with batch normalization between the last LSTM and first FC layers. ReLU activation is applied after the first FC layer and sigmoid after the second. Mean square error (MSE) is used as the loss function:

$$L_n = |\hat{s} - s|^2 \quad (3)$$

2.2. Iterative pruning and knowledge distillation

Pruning methods include structured pruning and unstructured pruning. Unstructured pruning can achieve smaller loss and higher pruning ratio than structure pruning. Therefore it is more suitable for hardware devices that can not be accelerated. Commonly used Pruning strategies include one-shot pruning [23] and iterative pruning [13]. One-shot pruning prunes the model once, resulting model lose too many parameters at one time, and the model after one-shot pruning can't be fine-tuning well. Iterative pruning is gradually pruning, which means the number of the parameters reduce slowly each time, and the model will be fine-tuned well in the end. Therefore, this paper use iterative pruning strategy to compress model. In addition, in order to increase the pruning ratio without degrading performance, l_1 regularization is adopted:

$$\Gamma_l = \frac{\lambda}{g(\varpi)} \sum_{\omega \in \varpi} |\omega| \quad (4)$$

where, ϖ denotes the set of all nonzero weights, and λ is a predefined hyperparameter, respectively. $g(\cdot)$ is cardinality of a set. The new loss function can be described as:

$$L_t = L_n + \Gamma_l \quad (5)$$

First, to determine the proportion of pruning, we analyze the sensitivity of each weight tensor. Second, unstructured pruning is carried out according to the preset pruning rate. Third knowledge distillation is used to fine-tune the model. The vanilla model is usually used as the teacher, but it can make a big gap between the teacher model and the student model.

Algorithm 1: Iterative pruning and knowledge distillation

- Data:**
1. ϖ_j : Indicates all non-zero weights in j -th weight tensor $W_j, \forall j$.
 2. v : Dataset used for validation.
 3. ζ : Controllable loss threshold.
 4. $\eta(v, \theta)$: Indicates the loss obtained by input validation set v into the network with nonzero parameter θ .
 5. $f'_\theta(\cdot), f_\theta(\cdot)$: $f'_\theta(\cdot)$ indicates the network after each pruning, $f_\theta(\cdot)$ indicates the network before each pruning.
 6. ρ_j : pruning ratio for j -th weight tensor.

Result: Network after pruning and fine-tuning

```

1 for each iteration  $i$  do
2   for each weight tensor  $W_j$  do
3      $\rho_j = 0\%$ 
4     Let  $\varphi \subseteq \varpi$  be the set of  $\rho\%$  of the smallest
5     absolute value of non-zero weight in each
6     weight tensor  $W_j$ 
7      $s \leftarrow \eta(v, \theta | \omega = 0, \forall \omega \in \varphi) - \eta(v, \theta)$ 
8     if  $s < \zeta$  then
9       |  $\rho_j = \rho_j + 5\%$ 
10    else
11      | break;
12    end
13  end
14  Get  $\rho_j$  for weight tensor  $W_j, \forall j$ .  $\rho_j$  is used to
15  prune the network. When the pruning is done,
16  network is fine-tuned by knowledge distillation.
17  The loss function is described as :
18   $L = L_n + L_d + \Gamma_l$ 
19 end
20 return: Network after pruning and fine-tuning

```

Leading to a poor convergence for the student model. To reduce the gap, after one time pruning, we adopt the model before each pruning as the teacher model and the model after each pruning as the student model. Using the teacher model to fine-tune the student model. The procedure is shown in Figure 1. The loss function for distillation is:

Table 1: Performance of GCRN after 1 to 5-th pruning and quantization.

| Metrix | STOI | | | | | | | | PESQ | | | | | | | | Model Size |
|--------------|--------|------|------|------|-----------|------|------|------|--------|------|------|------|-----------|------|------|------|------------|
| | Babble | | | | Cafeteria | | | | Babble | | | | Cafeteria | | | | |
| Test SNR | -5db | 0db | 5db | AVG | -5db | 0db | 5db | AVG | -5db | 0db | 5db | AVG | -5db | 0db | 5db | AVG | |
| Mixture | 58.4 | 71.4 | 82.9 | 70.9 | 56.8 | 71.1 | 82.0 | 70.0 | 1.50 | 1.77 | 2.08 | 1.78 | 1.35 | 1.73 | 2.03 | 1.70 | |
| Baseline | 80.2 | 90.1 | 94.4 | 88.2 | 77.2 | 89.2 | 93.5 | 86.6 | 2.06 | 2.61 | 3.03 | 2.57 | 1.97 | 2.51 | 2.93 | 2.47 | |
| Iter1 | 80.2 | 90.1 | 94.5 | 88.3 | 77.1 | 89.1 | 93.5 | 86.6 | 2.03 | 2.58 | 3.01 | 2.54 | 1.96 | 2.47 | 2.90 | 2.45 | |
| Iter2 | 80.2 | 90.1 | 94.5 | 88.3 | 77.0 | 89.2 | 93.5 | 86.6 | 2.04 | 2.60 | 3.01 | 2.55 | 1.97 | 2.50 | 2.92 | 2.46 | |
| Iter3 | 80.0 | 90.1 | 94.4 | 88.2 | 77.0 | 89.2 | 93.5 | 86.6 | 2.03 | 2.59 | 3.01 | 2.54 | 1.97 | 2.50 | 2.91 | 2.46 | |
| Iter4 | 79.7 | 90.0 | 94.4 | 88.1 | 76.6 | 89.0 | 93.4 | 86.4 | 2.02 | 2.59 | 3.01 | 2.54 | 1.96 | 2.49 | 2.91 | 2.45 | |
| Iter5 | 79.7 | 89.8 | 94.3 | 88.0 | 76.7 | 88.9 | 93.4 | 86.3 | 2.03 | 2.58 | 3.00 | 2.54 | 2.00 | 2.50 | 2.91 | 2.46 | |
| Quantization | 78.3 | 89.0 | 93.8 | 87.1 | 75.4 | 88.3 | 92.9 | 85.5 | 1.98 | 2.55 | 2.97 | 2.50 | 1.94 | 2.48 | 2.88 | 2.44 | |

$$L_d = \alpha |f'_\theta(m)_i - f_\theta(m)_i|^2 \quad (6)$$

where $f'_\theta(\cdot)_i$ indicates the network after the i -th pruning, $f_\theta(\cdot)_i$ indicates the network before each pruning, α is predefined weighting parameters, respectively. Therefore, the final loss can be defined as:

$$L = L_t + L_d + \Gamma_l \quad (7)$$

Specific details are described in Algorithm 1.

2.3. Clustering-based quantization

After iterative pruning and knowledge distillation, the quantitative strategy based on clustering is adopted to further compress the model. The essence of quantification is to use K-means to cluster the weights of each layer. After K-means converge, weights belonging to the same cluster share the same weight. We divide n original weights $W = \{w_1, w_2, w_3 \dots w_n\}$ into k clusters $c = \{c_1, c_2, c_3 \dots c_k\}$. The original weights are approximated by the cluster centers. A text is created to store the cluster centers for each weight tensor, in which each weight is tied to the corresponding cluster index. During inference, the value of each weight is looked up in the text. A 2-bit example is shown in Figure. 2. After sharing mechanism, the model only needs to store the corresponding index, and the storage space occupied by the model is greatly reduced. The compression ratio can be expressed as:

$$r = \frac{nb}{n \log_2 k + kb} \quad (8)$$

where n indicates the number of non-zero weights, b indicates the number of stored bits, k is cluster number, and $\log_2 k$ indicates the coded index, respectively.

The cluster quantization strategy used in this paper is similar to the pruning strategy. Each weight tensor determines a cluster center as small as possible according to the quantification strategy in this paper. To a certain extent, the performance loss caused by quantization can be reduced by using different number clustering centers for each weight tensor. On the other hand, the network quantization is flexible with the use of cluster strategy. Refer to [14] for more details.

3. Experimental Results

3.1. Datasets

WSJ0 SI-84 [24], which includes 7138 utterances from 83 speakers (42 males and 41 females), is used to evaluate the proposed method. 77 speakers are used for training and remaining 6 are used for evaluation. For the training set, 10000 noises

from a sound effect library (available at <https://www.sound-ideas.com>) [25] are used. Specifically, we mix a randomly selected training utterance with a random cut from the 10000 noises. The SNR is randomly sampled between -5 and 0 dB. The training set includes 300,000 mixtures. Then, creating 30,000 mixtures as validate set by using the same method. For the test set, the two challenging noises (babble and cafeteria) from an Auditec CD (available at <http://www.auditec.com>) are used to generate 300 mixtures at each SNR of -5, 0, and 5dB.

The perceptual evaluation of speech quality (PESQ) [26] and short-term objective intelligibility (STOI) [27] are used to evaluate speech quality and intelligibility, respectively. STOI values typically range from 0 to 1, which can be roughly interpreted as percent correct. PESQ values range from -0.5 to 4.5.

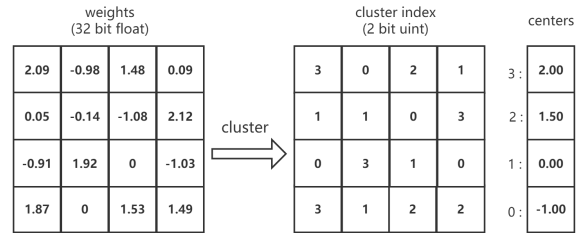


Figure 2: clustering-based quantization.

3.2. Experimental setup

For all experiments, we take a piece of speech through the sampling rate of 16kHz. The GCRN is trained in the frequency domain and according to the configuration described in [22]. For LSTM, we use 320 points FFT to get 161-D spectral features, then through a Mel-spectrogram transform matrix, the speech is mapped to a 128-dimensional mel space as input. The each pruning controllable loss threshold ζ is set empirically to 0.04, and the number of iteration i is set to 5, respectively. The distillation fine-tuning ratio α is set to 0.1. The configuration of quantization is the same as [14].

3.3. Experimental results

3.3.1. Effectiveness of knowledge distillation

Typically, pruning operations degrade model performance. The purpose of knowledge distillation in this paper is to fine-tune the performance of the model after pruning. To verify its effectiveness, some experiments are conducted as follows. Proposed

Table 2: Performance of LSTM after 1 to 5-th pruning and quantization.

| Metrix | STOI | | | | | | | | PESQ | | | | | | | | Model Size |
|--------------|--------|------|------|------|-----------|------|------|------|--------|------|------|------|-----------|------|------|------|------------|
| | Babble | | | | Cafeteria | | | | Babble | | | | Cafeteria | | | | |
| Test SNR | -5db | 0db | 5db | AVG | -5db | 0db | 5db | AVG | -5db | 0db | 5db | AVG | -5db | 0db | 5db | AVG | |
| Mixture | 58.4 | 71.4 | 82.9 | 70.9 | 56.8 | 71.1 | 82.0 | 70.0 | 1.50 | 1.77 | 2.08 | 1.78 | 1.35 | 1.73 | 2.03 | 1.70 | |
| Baseline | 72.8 | 85.4 | 91.9 | 83.4 | 71.2 | 84.6 | 90.9 | 82.2 | 1.77 | 2.28 | 2.71 | 2.25 | 1.81 | 2.30 | 2.66 | 2.26 | 3.71MB |
| Iter1 | 73.2 | 85.4 | 91.9 | 83.5 | 71.1 | 84.6 | 90.9 | 82.2 | 1.77 | 2.28 | 2.71 | 2.25 | 1.81 | 2.30 | 2.65 | 2.25 | 2.06MB |
| Iter2 | 73.2 | 85.4 | 91.9 | 83.5 | 71.1 | 84.6 | 90.9 | 82.2 | 1.78 | 2.28 | 2.71 | 2.26 | 1.81 | 2.30 | 2.66 | 2.26 | 1.98MB |
| Iter3 | 73.2 | 85.4 | 91.9 | 83.5 | 71.0 | 84.6 | 90.0 | 81.9 | 1.78 | 2.28 | 2.71 | 2.26 | 1.81 | 2.30 | 2.66 | 2.26 | 1.89MB |
| Iter4 | 73.3 | 85.4 | 91.9 | 83.5 | 70.9 | 84.6 | 90.9 | 82.1 | 1.78 | 2.28 | 2.71 | 2.26 | 1.81 | 2.30 | 2.66 | 2.26 | 1.78MB |
| Iter5 | 73.3 | 85.3 | 91.9 | 83.5 | 70.9 | 84.6 | 90.9 | 82.1 | 1.78 | 2.28 | 2.71 | 2.26 | 1.81 | 2.30 | 2.65 | 2.25 | 1.57MB |
| Quantization | 73.1 | 85.3 | 91.8 | 83.4 | 70.9 | 84.6 | 90.8 | 82.1 | 1.77 | 2.27 | 2.70 | 2.25 | 1.79 | 2.29 | 2.64 | 2.24 | 0.33MB |

distillation to fine-tune the model (1). True label to fine-tune directly the model (2). A tiny GCRN model (3) is used as the baseline to verify the effectiveness of the pruning algorithm, where the weights number of tiny GCRN is the same as the pruned vanilla GCRN. Proposed distillation to fine-tune model referred to as Teacher and, true label to fine-tune the model referred to as No-teacher. Reducing the number of channels of the model referred to as Reduce-channel.

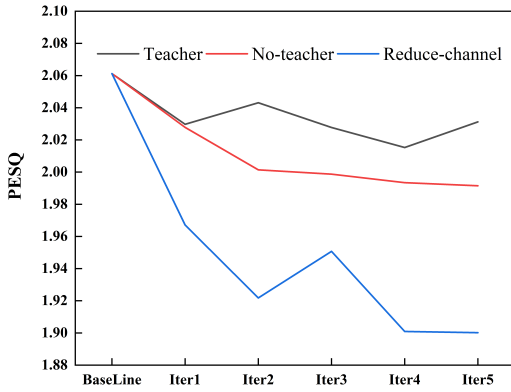


Figure 3: PESQ scores for -5 dB SNR from 1 to 5-th pruning

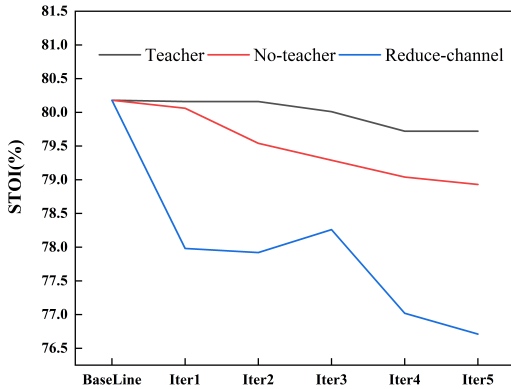


Figure 4: STOI scores for -5 dB SNR from 1 to 5-th pruning

The STOI and PESQ results at -5 dB SNR are shown in Figure 3 and 4, respectively. Iter1 to Iter5 represent pruning 1 to 5-th iterations, respectively. The results show that models fine-tuned by distillation have significantly improved performance than models without distillation. And as the iteration increases, the performance gap becomes larger. For example, after 5 it-

erations, the STOI of the teacher is 79.7, which is 0.8 and 3.0 better than the Non-teacher and Reduce-channel baselines. The experimental results prove that the model before pruning is useful, and it is effective to use knowledge distillation to fine-tune the pruned model. It also proves that pruning a large model is better than training a small model directly.

3.3.2. Validity of the compression algorithm

To verify the effectiveness of the compression algorithm, we compress GCRN and LSTM networks, respectively.

Table 1 and 2 show the performance of GCRN and LSTM networks after 5 iterations pruning and quantization at different SNR. Quantized performance is referred to as Quantization. As can be seen from Table 1, the model size for vanilla and 5-th iterations model is 37.27MB and 0.87MB, respectively, which means the compression rate is 40x, and the enhancement performance degrade rarely. From Table 3, the LSTM is compressed to 10x, and the performance is the same as before compression. The effectiveness of the proposed compression algorithm is proved by compress two networks.

Table 3: Model size of GCRN pruned 5 times, the unit is MB.

| | baseline | Iter1 | Iter2 | Iter3 | Iter4 | Iter5 |
|------------|----------|-------|-------|-------|-------|-------|
| Teacher | 37.27 | 11.56 | 8.10 | 6.43 | 5.68 | 4.96 |
| No-teacher | 37.27 | 11.56 | 8.16 | 6.48 | 5.76 | 5.00 |

3.3.3. Influence of distillation on model size

Table 3 presents the size model after each pruning by distillation fine-tune, and non-distillation fine-tune. The results show that the model whose performance is fine-tuned by distillation obtained a higher pruning rate in the next pruning. This is because, with distillation to fine-tune, the model has better performance before the next pruning. For the same ζ , more non-zero weights can be removed. It suggests that using distillation to fine-tune the model not only achieves higher performance but also obtains a smaller model than the without distillation.

4. Conclusions

In this paper, knowledge distillation, iterative pruning, quantization are combined to compress model. After each pruning, we use the model before pruning as the teacher model to restore the student model after pruning by combining the outputs of the teacher model with the true labels. The experiments show that the proposed method has good performance in speech enhancement and model compression ratio.

5. References

- [1] P. C. Loizou, *Speech enhancement: theory and practice*. CRC press, 2007.
- [2] S. Wisdom, J. R. Hershey, K. Wilson, J. Thorpe, M. Chinen, B. Patton, and R. A. Saurous, "Differentiable consistency constraints for improved deep speech enhancement," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 900–904.
- [3] S. Pascual, A. Bonafonte, and J. Serra, "Segan: Speech enhancement generative adversarial network," *arXiv preprint arXiv:1703.09452*, 2017.
- [4] J. Cheng, P.-s. Wang, G. Li, Q.-h. Hu, and H.-q. Lu, "Recent advances in efficient computation of deep convolutional neural networks," *Frontiers of Information Technology & Electronic Engineering*, vol. 19, no. 1, pp. 64–77, 2018.
- [5] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," *arXiv preprint arXiv:1710.09282*, 2017.
- [6] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.
- [7] Y. Gong, L. Liu, M. Yang, and L. Bourdev, "Compressing deep convolutional networks using vector quantization," *arXiv preprint arXiv:1412.6115*, 2014.
- [8] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.
- [9] A. Novikov, D. Podoprikin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," *Advances in neural information processing systems*, vol. 28, 2015.
- [10] V. Klema and A. Laub, "The singular value decomposition: Its computation and some applications," *IEEE Transactions on automatic control*, vol. 25, no. 2, pp. 164–176, 1980.
- [11] Y. LeCun, J. Denker, and S. Solla, "Optimal brain damage," *Advances in neural information processing systems*, vol. 2, 1989.
- [12] M. S. Zhang and B. Stadie, "One-shot pruning of recurrent neural networks by jacobian spectrum evaluation," *arXiv preprint arXiv:1912.00120*, 2019.
- [13] G. Castellano, A. M. Fanelli, and M. Pelillo, "An iterative pruning algorithm for feedforward neural networks," *IEEE transactions on Neural networks*, vol. 8, no. 3, pp. 519–531, 1997.
- [14] K. Tan and D. Wang, "Compressing deep neural networks for efficient speech enhancement," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 8358–8362.
- [15] V. Lebedev and V. Lempitsky, "Fast convnets using group-wise brain damage," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2554–2564.
- [16] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," *Advances in neural information processing systems*, vol. 28, 2015.
- [17] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.
- [18] Z. Huang and N. Wang, "Like what you like: Knowledge distill via neuron selectivity transfer," *arXiv preprint arXiv:1707.01219*, 2017.
- [19] Z. Xu, Y.-C. Hsu, and J. Huang, "Training shallow and thin networks for acceleration via knowledge distillation with conditional adversarial networks," *arXiv preprint arXiv:1709.00513*, 2017.
- [20] X. Hao, S. Wen, X. Su, Y. Liu, G. Gao, and X. Li, "Sub-band knowledge distillation framework for speech enhancement," *arXiv preprint arXiv:2005.14435*, 2020.
- [21] S. Nakaoka, L. Li, S. Inoue, and S. Makino, "Teacher-student learning for low-latency online speech enhancement using wave-u-net," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 661–665.
- [22] K. Tan and D. Wang, "Learning complex spectral mapping with gated convolutional recurrent networks for monaural speech enhancement," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 380–390, 2019.
- [23] T. Lin, S. U. Stich, L. Barba, D. Dmitriev, and M. Jaggi, "Dynamic model pruning with feedback," *arXiv preprint arXiv:2006.07253*, 2020.
- [24] D. B. Paul and J. Baker, "The design for the wall street journal-based csr corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York, February 23-26, 1992*, 1992.
- [25] J. Chen, Y. Wang, S. E. Yoho, D. Wang, and E. W. Healy, "Large-scale training to increase speech intelligibility for hearing-impaired listeners in novel noises," *The Journal of the Acoustical Society of America*, vol. 139, no. 5, pp. 2604–2612, 2016.
- [26] 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 *2001 IEEE international conference on acoustics, speech, and signal processing. Proceedings (Cat. No. 01CH37221)*, vol. 2. IEEE, 2001, pp. 749–752.
- [27] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time-frequency weighted noisy speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 7, pp. 2125–2136, 2011.