2024 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, SEPT. 22–25, 2024, LONDON, UK

INVESTIGATING PASSIVE FILTER PRUNING FOR EFFICIENT CNN-TRANSFORMER AUDIO CAPTIONING

Xuenan Xu[1], *Arshdeep Singh*[2], *Mengyue Wu*[1], *Wenwu Wang*[2], *Mark D. Plumbley*[2]

[1]MoE Key Lab of Artificial Intelligence X-LANCE Lab, Shanghai Jiao Tong University, China
[2]Centre for Vision, Speech and Signal Processing (CVSSP), University of Surrey, UK

ABSTRACT

Although automated audio captioning (AAC) has achieved remarkable performance improvement in recent years, concerns about the complexity of AAC models have drawn little attention from the research community. To reduce the number of model parameters, passive filter pruning has been successfully applied to convolution neural networks (CNNs) in audio classification tasks. However, due to the discrepancy between audio classification and AAC, these pruning methods are not necessarily suitable for captioning. In this work, we investigate the effectiveness of several passive filter pruning approaches on an efficient CNN-Transformer-based AAC architecture. Through extensive experiments, we find that under the same pruning ratio, pruning from the later convolution blocks significantly improves the performance. Utilizing the norm-based pruning method, our pruned model reduces the parameter number by 15% compared to that of the original model while maintaining a similar performance.

Index Terms— CNNs, audio captioning, pruning filters, EfficientNet

1. INTRODUCTION

Automated Audio Captioning (AAC) aims to generate descriptive text for a given audio clip. This technology has many potential applications, such as automatic summarization of multimedia content on the Internet and assisting hearing-impaired individuals in perceiving the world. In addition, AAC can serve as a data augmentation technique for audio-text retrieval [1] and text-to-audio generation tasks [2]. Significant advancements have been achieved in the field of AAC recently, in terms of accuracy [3], diversity [4], and level of details in audio descriptions [5]. The AAC task in the Detection and Classification of Acoustic Scenes and Events (DCASE) challenge has attracted substantial interest from researchers, further propelling the field forward.

Although the state-of-the-art (SOTA) performance of AAC has been improved constantly in recent years, these models are typically of large size. For example, the parameter number of HTSAT-BART [6] is over 170 million. Such large-scale models encompass a significant number of parameters and extensive computational cost, which poses considerable challenges in their deployment for resource-constrained environments. As a matter of fact, studies have indicated that many parameters in large models may be redundant for specific tasks [7]. However, the compression and pruning of audio captioning models remain underexplored in current works.

Model compression involves eliminating elements such as individual weights or whole filters contributing least to performance from the original model [8]. Eliminating individual weights produces an unstructured sparse network that requires specialized software or hardware for speed-up [9]. On the other hand, eliminating the whole filter produces a structured network, where cross-platform inference is supported in off-the-shelf libraries. Therefore, filter pruning methods are advantageous compared to weight pruning methods. Most filter pruning methods are data-dependent or active, where the importance of filters (i.e., the contribution to the final result) is measured using a training dataset. Such active filter pruning methods either involve joint optimization of networks with filter importance computation [10, 11] or use filter outputs to measure importance [12, 13]. In contrast, passive pruning methods are data-independent as they directly compute importance using trained filter weights [14, 15]. In the audio classification domain, some works focus on passive filter pruning, mostly for audio tagging and audio scene classification tasks. For example, similarity-based passive filter pruning methods [16, 17] have been applied for audio scene classification tasks to eliminate redundant filters from CNNs. However, pruning has drawn little attention in the context of AAC. The effectiveness of these pruning methods is yet to be studied for AAC.

In this work, we focus on pruning audio captioning models. We employ a CNN-Transformer-based encoder-decoder architecture, with an EfficientNet-based CNN encoder. We explore various passive filter pruning methods for EfficientNet to assess their effectiveness for AAC. Since EfficientNet comprises multiple blocks, we can apply pruning from any block. We investigate the influence of the starting block for pruning and the tradeoff between pruned model size and

Table 1. The architecture of EfficientNet-B2. "Conv" and "BN" denote the standard 2D convolution and batch normalization layers, respectively. "MBConvBlock" is a MobileNet-based convolution block.

Block Index	Architecture	# Channel	# Params
1	Conv + BN	32	417
2	2 x MBConvBlock	16	2224
3	3 x MBConvBlock	24	29K
4	3 x MBConvBlock	48	106K
5	4 x MBConvBlock	88	429K
6	4 x MBConvBlock	120	874K
7	5 x MBConvBlock	208	3.07M
8	2 x MBConvBlock	352	2.75 M
9	Conv + BN	1408	501K

captioning performance. Our experiments indicate that with a similar number of parameters, pruning from later blocks yields significantly better performance. Among the pruning methods explored, norm-based ones are shown to be more effective. By applying pruning from the penultimate block, we manage to prune 15% parameters of the original model, while essentially maintaining its performance.

2. CNN-TRANSFORMER BASED AUDIO CAPTIONING FRAMEWORK

Our captioning model adopts an encoder-decoder framework. For an input audio $\mathbf{x}$, the encoder transforms it into a compact latent representation $\mathbf{f}$. Then the decoder predicts the probabilities of words over a vocabulary at each time step n, given previous ground truth words (during training) or predicted words (during inference) in an auto-regressive manner.

$$\mathbf{f} = \text{Enc}_\theta(\mathbf{x}) \tag{1}$$

$$p_n = \text{Dec}_\Phi(\mathbf{f}, w_1, w_2, \ldots, w_{n-1}) \tag{2}$$

where θ and Φ denote encoder and decoder parameters, respectively. $p_n \in [0,1]^{|\mathcal{V}|}$ denotes the predicted word probability, and $\mathcal{V}$ is the vocabulary. The final predicted caption is obtained by further applying the decoding algorithms on p_n, such as beam search decoding or nucleus sampling.

Specifically, our model adopts a CNN-based encoder and a Transformer-based decoder. The encoder and decoder architectures are presented in Section 2.1 and Section 2.2. We focus on pruning the CNN encoder using passive filter pruning methods. Then, Section 2.3 illustrated our fine-tuning approach to adapt the pruned model.

2.1. EfficientNet Encoder

We adopt the EfficientNet-B2 [18] as the encoder for its outstanding performance on the HEAR benchmark [19] and compact size. It takes a 2D mel-spectrogram as the input

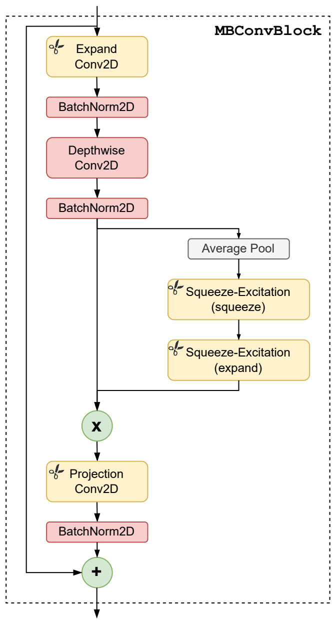


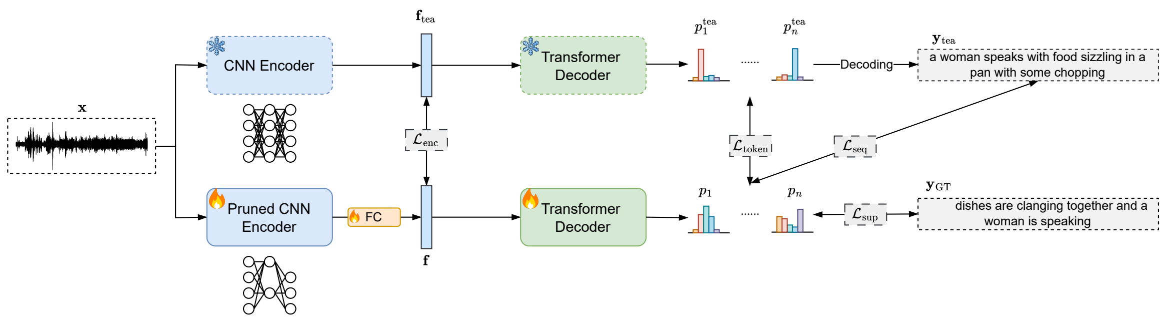
Fig. 1. The detailed structure of MBConvBlock, where activation functions are omitted as they do not contain parameters.

and encodes it into $\mathbf{f} \in \mathbb{R}^{T \times E}$, where T is the downsampled timesteps of the latent representation and E is the dimension of the embedding. The detailed structure of EfficientNet-B2 is shown in Table 1. It is divided into nine consecutive blocks. Except for the first and last block, which each consists of a convolution layer and a batch normalization layer, each block contains several MobileNet-based convolution blocks (MBConvBlock). The seventh and eighth blocks account for a large proportion of the overall parameters ($\sim$ 75%). The MBConvBlock structure is shown in Figure 1. Since the first few blocks contain few parameters, we only prune the latter blocks. In this setting, we denote the index of the block that pruning starts as *pruning starting block*. The impact of changing the pruning starting blocks on the performance will be investigated in Section 5.1.

We focus on pruning the convolution layers in the block. Since the input and output channel numbers of the depthwise convolution layer are the same, we only prune the remaining convolution layers, as marked by scissors symbols in the figure. The rest parts are affected by corresponding pruned layers though these layers themselves are not pruned directly. For example, as the first convolution layer is pruned, the batch normalization layer after it should adjust accordingly since the output channel number is reduced.

2.2. Transformer Decoder

For the decoder, we adopt a simple two-layer Transformer decoder as it performs well in previous challenges [20] with



$\mathbf{f}_{\text{tea}}$
CNN Encoder
Transformer Decoder
p_1^{tea}
p_n^{tea}
Decoding
$\mathbf{y}_{\text{tea}}$
a woman speaks with food sizzling in a pan with some chopping
$\mathbf{x}$
$\mathcal{L}_{\text{enc}}$
$\mathcal{L}_{\text{token}}$
$\mathcal{L}_{\text{seq}}$
Pruned CNN Encoder
FC
Transformer Decoder
p_1
p_n
$\mathcal{L}_{\text{sup}}$
$\mathbf{y}_{\text{GT}}$
dishes are clanging together and a woman is speaking
$\mathbf{f}$

Fig. 2. The knowledge distillation framework for adapting the pruned network. The frozen teacher provides supervision signals for the trainable student. We combine a series of loss functions, stemming from various supervision signals for training, including encoder-level ($\mathcal{L}_{\text{enc}}$), token-level ($\mathcal{L}_{\text{token}}$), sequence-level ($\mathcal{L}_{\text{seq}}$) and supervised loss ($\mathcal{L}_{\text{sup}}$).

a small number of parameters. A fully-connected (FC) layer transforms the last hidden representation to the predicted word probabilities p. The whole EfficientNet-Transformer model has 12.4 million parameters.

2.3. Fine-tuning Pruned Model by Knowledge Distillation

Before pruning, we first use the teacher-student learning technique to learn an efficient but strong CNN-Transformer model from a large teacher model. Then, we prune the CNN encoder further to reduce its size. To fine-tune the pruned model, we use the knowledge distillation technique again, with the unpruned model as the teacher and the pruned model as the student.

Since both distillation processes adopt the same framework, we take the distillation of the pruned model as an example for illustration. The framework is shown in Figure 2. In this case, the teacher model is the unpruned one while the student model is the pruned version. Inspired by [21], we adopt loss functions calculated from various supervision signals to enhance distillation. For an input audio clip $\mathbf{x}$, latent representations ($\mathbf{f}_{\text{teacher}}$, $\mathbf{f}$) and predicted word probabilities (p^{teacher}, p) are obtained by the teacher and student models. Then the encoder-level loss is calculated as:

$$\mathcal{L}_{\text{enc}} = ||\mathbf{f}_{\text{tea}} - \text{FC}(\mathbf{f}(\theta; \mathbf{x}))||^2, \tag{3}$$

where FC denotes a fully connected projection layer that maps the student embedding to be as close to the teacher embedding as possible. The token-level loss is also used to make the student-predicted probabilities of ground truth caption $\mathbf{y}_{\text{GT}} = \{w_1^{\text{GT}}, w_2^{\text{GT}}, \ldots, w_{N_{\text{GT}}}^{\text{GT}}\}$ close to the teacher predicted one:

$$\mathcal{L}_{\text{token}} = \sum_{n=1}^{N_{\text{GT}}} \text{KL}(p_n(\theta, \Phi; \mathbf{x}) || p_n^{\text{tea}}), \tag{4}$$

In addition, teacher-predicted word probabilities can be further decoded into the caption $\mathbf{y}_{\text{tea}} = \{w_1^{\text{tea}}, w_2^{\text{tea}}, \ldots, w_{N_{\text{tea}}}^{\text{tea}}\}$ to perform sequence-level distillation:

$$\mathcal{L}_{\text{seq}} = -\sum_{n=1}^{N_{\text{tea}}} \log\left(p_{n, w_n^{\text{tea}}}(\theta, \Phi; \mathbf{x})\right), \tag{5}$$

where $p_{n,m}$ denotes the predicted probability of the m-th token in the vocabulary, at the n-th timestep. The supervised loss is calculated based on the ground truth caption similarly:

$$\mathcal{L}_{\text{sup}} = -\sum_{n=1}^{N_{\text{GT}}} \log\left(p_{n, w_n^{\text{GT}}}(\theta, \Phi; \mathbf{x})\right). \tag{6}$$

The final distillation loss is the combination of the above loss functions:

$$\mathcal{L} = \mathcal{L}_{\text{sup}} + \mathcal{L}_{\text{seq}} + \mathcal{L}_{\text{enc}} + \mathcal{L}_{\text{token}}. \tag{7}$$

For learning the unpruned model via knowledge distillation, the process is almost the same. The only difference is that $\mathcal{L}_{\text{token}}$ is not utilized since the teacher and student models use different tokenizers.

3. PASSIVE FILTER PRUNING FOR CONVOLUTIONAL NEURAL NETWORKS

In this section, we introduce several passive filter pruning methods used in this work, which are applied parts with scissors symbols in Figure 1.

The importance of convolutional filters to quantify whether to retain in CNN is measured layer by layer. We use passive filter pruning methods to measure the convolutional filter importance. The passive filter pruning methods compute the importance of the j-th filter, I_j, in a given intermediate layer of CNNs after directly applying norm-based and similarity based methods on CNN filters. After eliminating convolutional filters based on a user-defined pruning ratio, the pruned network is re-trained to regain the most of the lost performance. An overview of norm-based and similarity based pruning methods is given below,

Norm-based Pruning Methods: In these methods, the criterion *"smaller-norm-less-important"* is used, where a filter is considered less important if the filter has low-norm. Such methods consider low-norm filters to be numerically less important in producing output. For example, Li et al. [14] use I_j equals to l_1-norm, $\|\mathbf{F}\|_1 = \sum_{i=1}^{\text{length}(\mathbf{F})} |\mathbf{F}_i|$ or an l_2-norm, $\|\mathbf{F}\|_2 = (\sum_{i=1}^{\text{length}(\mathbf{F})} \mathbf{F}_i^2)^{\frac{1}{2}}$ of the filters ($\mathbf{F}$) to quantify the filter importance.

He et al. [22] proposed a method that measures $I_j = \|\mathbf{F} - \mathbf{F}_{\text{GM}}\|_2$ as l_2-norm of the filters from the geometric median ($\mathbf{F}_{\text{GM}}$) of all filters as given in Equation (8), where a filter with low l_2-norm from the $\mathbf{F}_{\text{GM}}$ is considered relatively less important than others as it represents more commonality than other filters,

$$\mathbf{F}_{\text{GM}} = \underset{\mathbf{F}_{\text{GM}}}{\text{argmin}} \sum_{j \in [1,N]} \|\mathbf{F}_{\text{GM}} - \mathbf{F}_j\|_2. \tag{8}$$

However, l_1`-norm` or `GM` methods may ignore the redundancy in selecting important filters as only high-norm filters are considered as important. Singh et al. [15] used the operator norm (`opnorm`) of filters to eliminate the filters that produce the least significant output. The importance of each filter is measured as $I_j = \|\mathbf{F}_j\|_{\text{op}}$, where $\|\mathbf{F}_j\|_{\text{op}} = \sigma_1$ represents the most significant singular value of $\mathbf{F}_j$. In contrast to l_1`-norm` or `GM` methods, `opnorm` considers geometrical relationships by incorporating the operator norm of the filters in computing the filter importance.

Similarity-based Pruning Methods: These methods consider filters that produce similar output to others as "redundant". The redundancy among filters is determined by utilizing pairwise similarity measures [16] and graph-based centrality measures [23]. For example, Singh et al. [16] computed a pairwise `cosine` distance between filters and removed one filter from each of the similar filter pairs. Kim et al. [24] applied clustering to the filters and consider an important filter from each cluster while eliminating the remaining filters. King et al. [23] used `graph`-centrality methods to eliminate redundant filters. In this approach, a filter with a high centrality is considered less important than others as it represents commonality and can be replaced by other filters without substantially affecting the performance.

4. EXPERIMENTAL SETUP

4.1. Data and Hyper-parameters

In this work, we conduct experiments on AudioCaps [25]. We use EfficientNet-B2[1] pre-trained on AudioSet to initialize the audio encoder. In the first distillation stage, we train an efficient student from a strong teacher. We use HTSAT-BART from [6] as the teacher. In the second distillation stage, the student from the first stage is taken as the teacher while the pruned one is taken as the new student. For both stages, the model is trained for 25 epochs with a batch size of 32. The learning rate is linearly warmed up to 5×10^{-4} in the first 5 epochs and then exponentially reduced to 5×10^{-7}. Beam search decoding with a beam size of 3 is used during inference.

[1]https://github.com/RicherMans/HEAR2021_EfficientLatent

4.2. Evaluation Metrics

We adopt SPIDEr [26] and FENSE [27] to evaluate the captioning performance. For simplicity, metrics such as BLEU [28] that do not correlate well with human judgments are not included. We also report the parameter number to compare the model size.

5. RESULTS

5.1. Influence of Pruning Starting Block

We first investigate the influence of the pruning starting block on the performance of the pruned model. The result is shown in Figure 3, where we keep the overall pruning ratio at 50% while varying the pruning starting block. Since the last two blocks of the encoder contain only 26% parameters of the whole model, the pruning ratio of 50% cannot be achieved if pruning starts from the eighth or ninth block. When pruning

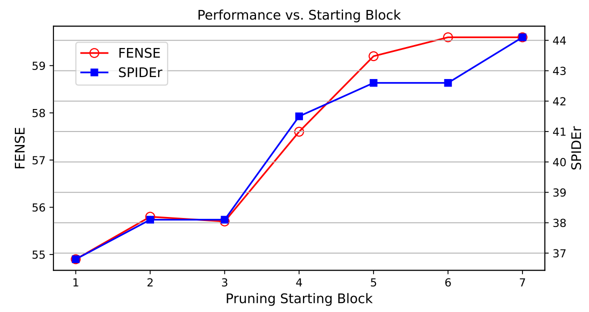


Fig. 3. The effect of different pruning starting blocks on the captioning performance, under the same pruning ratio of 50%.

starts from early blocks, each block is pruned with a small ratio. In contrast, pruning from later blocks results in each block being pruned with a larger ratio to maintain the same overall pruning ratio. The results indicate that with the same pruning ratio, pruning from later blocks leads to better performance. As Table 1 shows, the first few blocks contain very few parameters compared with later blocks. Therefore, pruning these blocks contributes little to the overall pruning ratio. Furthermore, the channel number of earlier blocks is smaller so each channel contains richer information than channels from later blocks. As a result, pruning filters of early blocks leads to significant performance degradation.

5.2. Size-Performance Tradeoff

We now explore the relationship between the captioning performance and the pruning ratio. We set the pruning starting block as 5 to evaluate the effect of a larger range of pruning ratios, from 10% to 65%. Figure 4 illustrates how the captioning performance changes as the pruning ratio becomes larger. The result indicates that a pruning ratio between 10% to 20% achieves a good tradeoff between model size and performance. Pruning ratios over 20% lead to a significant performance drop. We choose a pruning ratio of 15% in the following experiments.

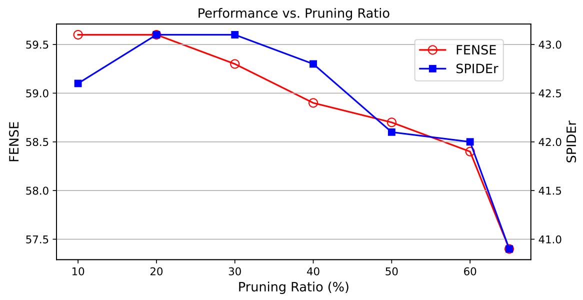


Fig. 4. The relationship between captioning performance and pruning ratio.

5.3. Influence of Pruning Methods

Finally, we compare the effectiveness of different passive filter pruning methods introduced in Section 3 on captioning. The performance of the original unpruned model is also reported for reference. The results are listed in Table 2. Although similarity-based methods generally outperform norm-based methods in audio classification [23], norm-based methods yield better results in captioning. This may be attributed to the fact that in the current captioning model, the CNN encoder does not directly output classification probabilities, but provides latent representations for the decoder to generate captions. Similarity-based methods eliminate filters that are close to other ones. However, the pruned filters may have a large impact on the encoded latent representation. In contrast, the filters pruned by norm-based methods are more likely to contribute less to the latent due to their smaller norm values. However, pruning methods do not influence the final performance significantly. The best-performing model achieves a performance comparable to the unpruned model with 85% parameters.

6. CONCLUSION

In this paper, we have presented passive filter pruning for the CNN-Transformer audio captioning framework. We employ various distillation loss functions to fine-tune the pruned model. Two types of pruning methods, norm-based and similarity-based methods, are evaluated on captioning. Our analysis reveals that under the same pruning ratio, pruning from later convolution blocks yields better results than pruning from very early blocks. The captioning performance under different pruning ratios shows that a pruning ratio of 10% $\sim$ 20% achieves a good tradeoff between performance and model size. The comparison of different pruning methods shows that norm-based methods generally perform better than similarity-based methods. Finally, the pruned model retains a performance comparable to the unpruned one with 85% parameters.

Table 2. The effect of different pruning methods on the captioning performance, with pruning starting from the eighth block and a pruning ratio of 15%.

Method		SPIDEr	FENSE
Unpruned		48.8	63.8
Norm-based	l_1-`norm`	**48.1**	**63.1**
	`GM`	47.7	**63.1**
	`opnorm`	46.9	62.8
Similarity-based	`cosine`	47.6	62.7
	`graph`	47.6	62.8

7. ACKNOWLEDGEMENT

This work was supported in part by Guangxi major science and technology project (No. AA23062062) and the Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/T019751/1 ”AI for Sound”. For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) license to any Author Accepted Manuscript version arising. This publication is supported by datasets that are openly available at locations referenced in this paper.

8. REFERENCES

[1] A.-M. Oncescu, A. Koepke, J. F. Henriques, Z. Akata, and S. Albanie, “Audio retrieval with natural language queries,” in *Proc. ISCA Interspeech*, 2021, pp. 2411–2415.

[2] G. Li, X. Xu, L. Dai, M. Wu, and K. Yu, “Diverse and vivid sound generation from text descriptions,” in *Proc. IEEE ICASSP*. IEEE, 2023, pp. 1–5.

[3] S.-L. Wu, X. Chang, G. Wichern, J.-w. Jung, F. Germain, J. L. Roux, and S. Watanabe, “BEATs-based audio captioning model with INSTRUCTOR embedding supervision and ChatGPT mix-up,” DCASE2023 Challenge, Tech. Rep., 2023.

[4] X. Mei, X. Liu, J. Sun, M. D. Plumbley, and W. Wang, “Diverse audio captioning via adversarial training,” in *Proc. IEEE ICASSP*, 2022, pp. 8882–8886.

[5] Z. Xie, X. Xu, M. Wu, and K. Yu, “Enhance temporal relations in audio captioning with sound event detection,” in *Proc. ISCA Interspeech*, 2023, pp. 4179–4183.

[6] X. Mei, C. Meng, H. Liu, Q. Kong, T. Ko, C. Zhao, M. D. Plumbley, Y. Zou, and W. Wang, “Wavcaps: A ChatGPT-assisted weakly-labelled audio captioning dataset for audio-language multimodal research,” *arXiv preprint arXiv:2303.17395*, 2023.

[7] J. Frankle and M. Carbin, “The lottery ticket hypothesis: Finding sparse, trainable neural networks,” in *Proc. ICLR*, 2018.

[8] T. Liang, J. Glossner, L. Wang, S. Shi, and X. Zhang, “Pruning and quantization for deep neural network acceleration: A survey,” *Neurocomputing*, vol. 461, pp. 370–403, 2021.

[9] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, “EIE: Efficient inference engine on compressed deep neural network,” *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 243–254, 2016.

[10] S. Lin, R. Ji, C. Yan, B. Zhang, L. Cao, Q. Ye, F. Huang, and D. Doermann, “Towards optimal structured CNN pruning via generative adversarial learning,” *Proc. CVPR*, pp. 2790–2799, 2019.

[11] J.-H. Luo and J. Wu, “AutoPruner: An end-to-end trainable filter pruning method for efficient deep model inference,” *Pattern Recognition*, vol. 107, p. 107461, 2020.

[12] M. Lin, R. Ji, Y. Wang, Y. Zhang, B. Zhang, Y. Tian, and L. Shao, “HRank: Filter pruning using high-rank feature map,” *Proc. CVPR*, pp. 1529–1538, 2020.

[13] S.-K. Yeom, K.-H. Shim, and J.-H. Hwang, “Toward compact deep neural networks via energy-aware pruning,” *arXiv preprint arXiv:2103.10858*, 2021.

[14] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, “Pruning filters for efficient ConvNets,” *Proc. ICLR*, 2017.

[15] A. Singh, H. Liu, and M. D. Plumbley, “E-PANNs: Sound recognition using efficient pre-trained audio neural networks,” in *Inter-Noise and Noise-Con Congress and Conference Proceedings*, vol. 268, no. 1. Institute of Noise Control Engineering, 2023, pp. 7220–7228.

[16] A. Singh and M. D. Plumbley, “A passive similarity based CNN filter pruning for efficient acoustic scene classification,” in *Proc. ISCA Interspeech*, 2022.

[17] ——, “Efficient similarity-based passive filter pruning for compressing cnns,” in *Proc. IEEE ICASSP*. IEEE, 2023, pp. 1–5.

[18] M. Tan and Q. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in *Proc. ICML*. PMLR, 2019, pp. 6105–6114.

[19] J. Turian, J. Shier, H. R. Khan, B. Raj, B. W. Schuller, C. J. Steinmetz, C. Malloy, G. Tzanetakis, G. Velarde, K. McNally *et al.*, “HEAR: Holistic evaluation of audio representations,” in *NeurIPS 2021 Competitions and Demonstrations Track*, 2022, pp. 125–145.

[20] X. Xu, Z. Xie, M. Wu, and K. Yu, “The SJTU system for DCASE2022 challenge task 6: Audio captioning with audio-text retrieval pre-training,” DCASE2022 Challenge, Tech. Rep., 2022.

[21] U. Cappellazzo, M. Yang, D. Falavigna, and A. Brutti, “Sequence-level knowledge distillation for class-incremental end-to-end spoken language understanding,” in *Proc. ISCA Interspeech*, 2023, pp. 2953–2957.

[22] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, “Filter pruning via geometric median for deep convolutional neural networks acceleration,” *Proc. CVPR*, pp. 4340–4349, 2019.

[23] J. A. King, A. Singh, and M. D. Plumbley, “Compressing audio CNNS with graph centrality based filter pruning,” in *2023 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2023, pp. 1–5.

[24] M. Park, W. Kim, and S. Kim, “REPrune: Filter pruning via representative election,” *arXiv preprint arXiv:2007.06932*, 2020.

[25] C. D. Kim, B. Kim, H. Lee, and G. Kim, “AudioCaps: Generating captions for audios in the wild,” in *Proc. NAACL*, 2019, pp. 119–132.

[26] S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, “Improved image captioning via policy gradient optimization of SPIDEr,” in *Proc. CVPR*, 2017, pp. 873–881.

[27] Z. Zhou, Z. Zhang, X. Xu, Z. Xie, M. Wu, and K. Q. Zhu, “Can audio captions be evaluated with image caption metrics?” in *Proc. IEEE ICASSP*, 2022, pp. 981–985.

[28] P. Kishore, R. Salim, W. Todd, and Z. Wei-Jing, “BLEU: a Method for Automatic Evaluation of Machine Translation,” in *Proc. ACL*, 2002, pp. 311–318.