LIGHT GATED MULTI MINI-PATCH EXTRACTOR FOR AUDIO CLASSIFICATION

Bo He[⋆]*, Shiqi Zhang*[⋆]*, Xianrui Wang*[⋆]*, Zheng Qiu*[⋆]*,*
Daiki Takeuchi[†]*, Daisuke Niizumi*[†]*, Noboru Harada*[†]*, Shoji Makino*[⋆]

[⋆] Waseda University, Japan [†] NTT Corporation, Japan

ABSTRACT

Audio classification, which serves as a fundamental step for acoustic signal processing, has attracted a lot of researchers interest, and numerous audio classification neural networks have been proposed. In these networks, pooling modules, which compress audio features, are essential due to their computational capacity. However, compressing the signal will inevitably cause the loss of relevant information. To mitigate this issue, a large number of parameters should be used. In this paper, we present a novel pooling method called gated multi mini-patch extractor (GMME), in which multiple convolution layers are used to extract relevant information at different aspects, including time frames, pseudo-frequency bins, and global features. Besides, the gate mechanism is adopted to emphasize the correlation among the original features. Several simulations demonstrate that, compared to the baseline, our method can achieve comparable or slightly better performance with a significant reduction in model size.

***Index Terms*—** Audio classification, feature extraction, pooling module, gated multi mini-patch extractor

1. INTRODUCTION

Audio events, to name a few, animal sounds, speech and music, are fundamental parts of our daily lives. As there are various types of audio events, audio classification should be carried out before other tasks such as enhancement, separation, label indexing, and segmentation. Traditional methods use Gaussian mixture model [1] or hidden Markov model [2] as classifiers, in which time-frequency representations, e.g., short-time fourier transformation (STFT) spectrogram and Mel spectrogram, are used as input features. Limited by computational capacity and the variety of datasets, the accuracy of these methods is far from satisfying.

With the boom of deep neural networks (DNNs), numerous audio event classification networks have been proposed. These models, especially those based on the convolutional neural network (CNN) [3, 4], recurrent neural network [5, 6], and transformer architectures [7–9], have achieved remarkable superiority over the aforementioned machine learning methods. Inspired by the great achievements of the computer vision (CV) field, a popular kind of audio classification method that regards the spectrogram of an audio signal as an image has been proposed [10–12]. In these methods, various DNNs are implemented to extract features from these spectrograms. Then the fully connected layer is carried out for final judgment. The main problem of such methods is that they use a fixed time-frequency representation which might not be able to extract the time-frequency structures of different kinds of audio [13–15].

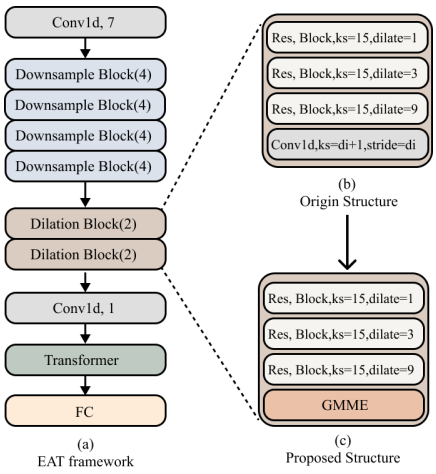

Fig. 1: Illustration for the EAT architecture and replacing in dilation block. (a) EAT architecture [16], (b) AA pooling module [17], (c) GMME module.

To address this problem, some end-to-end systems have been proposed [18, 19]. These systems generally consist of three major parts. Firstly, a CNN-based encoder is implemented to extract time-domain features as pseudo-frequency. This transformation can significantly improve feature extraction accuracy with some specific techniques [20, 21]. The second part entails pooling, which aims at shortening the duration of time frames and, consequently, reducing the dimension of features from the first stage. Finally, fully connected layers are carried out for classification. Although end-to-end methods are theoretically more flexible and are able to achieve better performance, there are still some problems, such as an enormous number of parameters or insufficient generalization [14]. To deal with these problems, the end-to-end audio transformer (EAT) [16], which employs extra data augmentation, to be specific, freqmix and phasemix [16] to expand data volume, consequently, getting better generalization.

For both of the aforementioned approaches, sufficient

This work was supported by JSPS KAKENHI Grant Number 23H03423 and China Scholarship Fund(202206060111).

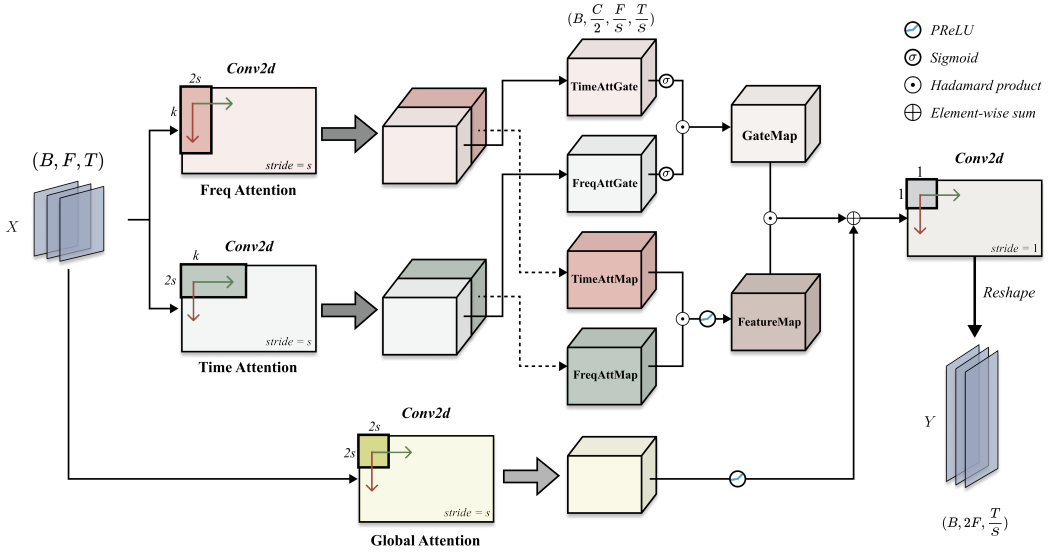

Fig. 2: Detailed structure of Gated Multi Mini-patch Extractor (GMME).

time domain pooling is essential due to the following three reasons. Firstly, Network should adopt pooling to achieve the purpose of reducing the computation of the model and retaining the main features, which is similar to a down-sampling operation. Secondly, insufficient pooling will lead to biased statistical estimates due to the natural non-stationarity of audio signals. Lastly, it should capture the periodic features of audio signals and consequently minimize to lose some key information. To mitigate these concerns, we presented a simplistic method named gated multi mini-patch extractor (GMME) for pooling time frames that allows for the interdependence of both temporal and frequency domain data with much fewer parameters compared to best technique in no pre-training situation (scratch). Mini-patch operations use rectangular convolution kernels to extract features, whose shape of kernel looks like the mini-size patch. EAT's small model (EAT-S) was considered the backbone network architecture for our study, wherein we trained and assessed our proposed approach using the ESC-50 [22], Speech Command [23] and UrbanSound8K [24] datasets as benchmarks. The mentioned datasets were used to evaluate the efficacy of our proposed method. Simulations validate that the proposed method successfully retains classification features while decreasing over 98% of the parameters compared to the original anti-alias pooling module.

2. RELATED WORK

2.1. Transformer

Transformer [25] model has exhibited exceptional proficiency in natural language processing [25, 26] and CV [27, 28]. The primary application of this technique involves extracting and converting relevant features with the manipulation of sequential data through a series of multi-head self-attention mechanisms. Because of its ability to extract long-term attention, transformer has also been implemented in audio classification. The audio spectrogram transformer (AST) [7] is a pure-transformer model based on the vision transformer [27] architecture. Some similar networks, such as HTS-AT [9], Beats [8] were also proposed.

Although transformer-based models have achieved promising performance, they may still disregard local prior knowledge. Consequently, these models frequently possess substantial parameters and necessitate prolonged training periods. Furthermore, real-world tasks are constrained by training datasets without the same data distribution, requiring models to be more robust. In order to solve these problems, researchers have leveraged the flexibility and efficacy of CNN and integrated them with transformer [6], and the best work in scratch is EAT [16]. Owing to its implementation of efficient data augmentation techniques, efficient dilated residual blocks broaden the perceptual field in the time domain, and anti-aliasing pooling is utilized to mitigate the distortion by down-sampling. The EAT model exhibits superior performance even when data is scarce.

2.2. Anti-alias (AA) pool

For mitigating aliasing caused by the pooling, EAT applies a low-pass filter with a blur kernel as pool and combines 1-d convolution to form downsample module, whose pool method is named Anti-alias pooling [17]. When the temporal receptive field is small, this operation reduces aliasing and enhances translational invariance. However, with the gradual increment of the pseudo-frequency domain resolution, the parameter size of the 1-d convolution grows and the temporal receptive field expands. Besides, the blur kernel might ignore the spectral correlation and alter the distribution of its features, leading to performance degradation [29].

Table 1: The results in ESC-50 and UrbanSound8K, N_k and N_c is respective Kernel size and Channel number. #P-Param is the trainable parameter number of the pooling module, while #W-Param represents the size of the whole model.

Model	Set	#P-Param(K)	#W-Param(M)	Accuracy(%)	
EAT-S-AA (conventional)		1961.1	5.18	ESC-50	Urban8k
				91.25	**84.11**
EAT-S-GMME (proposed)	$N_k = 32, N_c = 8$	17.4	3.23	90.80	83.64
	$N_k = 32, N_c = 16$	34.9	3.25	91.10	83.64
	$N_k = 64, N_c = 16$	**67.8**	**3.28**	**91.30**	83.59

3. PROPOSED METHOD

In this paper, we present the GMME to improve the preservation of pseudo-frequency and temporal features during the pooling process. We replace the anti-alias pooling module in EAT with our GMME, and the structure of the proposed network is illustrated in Fig. 1. GMME takes a tensor $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_B] \in \mathbb{R}^{f \times t}$ as input, and the corresponding output is $Y = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_B] \in \mathbb{R}^{2f \times \frac{t}{s}}$, where B is the batch size and s is the stride. Along the time frames, Y undergoes pooling, while along the pseudo-frequency bins, it expends. In audio convolution operations, it is often beneficial to increase the pseudo-frequency domain resolution for feature learning [30]. The detailed structure of GMME is described in Fig. 2, in which convolution parameters are kernel size and stride in order.

The main workflow of GMME can be divided into four steps. Firstly, temporal and pseudo-spectral relationships are encoded by local characteristics. In order to focus on neighboring information, we handle $\mathbf{x}_i$ into frequency, temporal flows, using rectangular convolution kernels for CNN's locality and anti-aliasing [30,31], and a global branch for shortcut. Specifically, given the input $\mathbf{x_i} \in \mathbb{R}^{f \times t}$, we use kernels with three spatial ranges: $[N_k, 2s]$, $[2s, N_k]$, and $[2s, 2s]$, where N_k denotes the kernel size and s represents the stride, i.e., the rate of the time down-sampling. Based on the physical properties of the spectrogram, features are effectively extracted by adopting non-square kernels for frequency and time, we call this operation mini-patch. At the same time, the channel of features was expanded, which quantity was represented as N_c. Once the GMME appropriately fixes the 2-d convolution kernel size N_k and channel extension N_c, its the parameter is stable and it can be adapted to input shape change. Instead of exponentially increasing on AA's 1-d convolutional layer with expansion of the pseudo-frequency domain. Mathematically, this process can be denoted as

$$\begin{aligned} \mathbf{Z}^f &= PRelu\left[\mathcal{F}_f(\mathbf{x}_i)\right], \\ \mathbf{Z}^t &= PRelu\left[\mathcal{F}_t(\mathbf{x}_i)\right], \\ \mathbf{Z}^g &= PRelu\left[\mathcal{F}_g(\mathbf{x}_i)\right], \end{aligned} \tag{1}$$

where $\mathcal{F}_f, \mathcal{F}_t, \mathcal{F}_g$ represent the convolutional operations in the frequency, temporal, and global branches, respectively. And $PRelu$ is the parametric rectified linear unit (PRelu) [32]. By applying these kernels, we obtain different attention features: $\mathbf{Z}^f, \mathbf{Z}^t \in \mathbb{R}^{N_c \times \frac{f}{s} \times \frac{t}{s}}$, and $\mathbf{Z}^g \in \mathbb{R}^{\frac{N_c}{2} \times \frac{f}{s} \times \frac{t}{s}}$.

Secondly, to extract the correlation between time and frequency features, we borrow the gate mechanism from the gated recurrent unit [33]. We divided the expanded channels $\mathbf{Z}^f$ and $\mathbf{Z}^t$ into two parts, one half of each is defined as gate maps, which are referred to as $\mathbf{G}^t$ and $\mathbf{G}^f$. The other half is treated as feature maps, denoted as $\mathbf{M}^t$ and $\mathbf{M}^\mathbf{f}$. This process can be expressed as (2).

$$\begin{aligned} \mathbf{G}^t &= \mathbf{Z}^t[:, : \frac{1}{2} \times N_c, :, :] \quad \mathbf{M}^t = \mathbf{Z}^t[:, \frac{1}{2} \times N_c :, :, :], \\ \mathbf{G}^f &= \mathbf{Z}^f[:, : \frac{1}{2} \times N_c, :, :] \quad \mathbf{M}^f = \mathbf{Z}^f[:, \frac{1}{2} \times N_c :, :, :]. \end{aligned} \tag{2}$$

Thirdly, gating is carried out to join time and frequency information and create the global gate weights map. The sigmoid function σ is used to convert $\mathbf{G}^t$ and $\mathbf{G}^f$ into weight maps over different domains. As strong probability distributions over different domains will emphasize the global weights and weak probability distributions will underplay the global weights, we utilize the Hadamard product $\odot$ in both weights maps to get the global gate weights map $\mathbf{G}$. The different domains will interact with each other to get the weights, acting as a mask for removing as much redundant or distracting information as possible and preserving useful information.

Similarly, $\mathbf{M}^t$ and $\mathbf{M}^f$ were also element-wise multiplied with each other to get the global feature $\mathbf{M}$, which is regarded as the time-frequency correlation feature map. This process can be described as (3) and (4)

$$\mathbf{G} = \sigma(\mathbf{G}^t) \odot \sigma(\mathbf{G}^f), \tag{3}$$

$$\mathbf{M} = PRelu(\mathbf{M}^t \odot \mathbf{M}^f). \tag{4}$$

Eventually, the Hadamard product of the gate weights map $\mathbf{G}$ and the feature map $\mathbf{M}$ is considered to provide feature fusion and global enhancement. To preserve information in the original representation, we employ a square kernel for global information learning as $\mathbf{Z}^g$. Then an element-wise sum with the weighted feature map, which works as a shortcut, is implemented. The output feature is then generated as

$$\mathbf{O} = reshape\Big\{\phi_c\big[\mathbf{Z}^g + (\mathbf{G} \odot \mathbf{M})\big]\Big\}. \tag{5}$$

where ϕ_c refers to the point-wise convolution kernel. It is used for channel adjustment and structure fitting. Besides, it also promotes fusion between the pseudo-frequency domain and the time domain. Finally, $reshape$ is implemented to adjust the feature dimensions for the rest of the modules.

Table 2: The results in Speech Commands V2, N_k and N_c is respective Kernel size and Channel number. #P-Param is the trainable parameter number of the pooling module, while #W-Param represents the size of the whole model.

Model	Set	#P-Param(K)	#W-Param(M)	Accuracy(%)
EAT-S-AA (conventional)		493.1	1.97	97.76
EAT-S-GMME (proposed)	$N_k = 32, N_c = 8$	17.4	1.50	97.74
	$N_k = 32, N_c = 16$	34.9	1.51	97.71
	$N_k = 64, N_c = 16$	**67.8**	**1.54**	**97.88**

4. EXPERIMENT

4.1. Dataset

- **ESC-50** [22]: It is a commonly used dataset for audio event classification, which consists of 2000 $5\,\mathrm{s}$-long audio signals from 50 different environmental audio events at a sampling rate of $44.1\,\mathrm{kHz}$. We down-sampled them to $22.05\,\mathrm{kHz}$, and trained 3500 epochs.
- **Speech Commands V2** [23]: Each audio sample in the dataset has a fixed duration of $1\,\mathrm{s}$ and a sampling frequency of $16\,\mathrm{kHz}$. The dataset includes 35 classification tasks, each representing a specific word or command to be recognized. Because there was the large amount of data with long training time with doing 10-folds experiments, we just trained for 300 epochs for comparison.
- **UrbanSound8K** [24]: This dataset comprises 8732 audio recordings, each belonging to one of 10 predefined classes, representing various common urban sounds. We resampled them to $22.05\,\mathrm{kHz}$, zero-padding the short samples to $4\,\mathrm{s}$. As the same limitation, we just trained 1500 epochs.

4.2. Training Configuration

We adopted the EAT-S model structure as backbone, which is renamed EAT-S-AA and replaced the conventional anti-alias pooling module in Dilation blocks with our GMME module, donated as EAT-S-GMME. We compared the performance and complexity of our network with the EAT-S-AA in the above three datasets. We also investigate the impacts of convolutional kernel sizes and channel widths.

All hyperparameters remain consistent with the open-source code[1] of the original paper [16], we also trained this code for comparison. On ESC-50 and UrbanSound8K datasets, the number of Res.blocks in Downsample Block is specified as four, and in Speech Comments V2 is three, which depend on different sample lengths. The parameters are optimized using the AdamW optimizer [34]. We use the Onecycle scheduler [35] with a maximum learning rate of $3e^{-4}$ and an epsilon of $1e^{-8}$, and employ the exponential moving average (EMA) [36, 37] method with a decay rate of 0.995. The batch size B is set to be 128 and stride s as 2. The loss is label-smoothing with a noise parameter set to be 0.1 for label classification tasks. We perform some label preserving and label mixing augmentations according to the methods described in the original paper [16] such as shifting, masking, noise, mixing [38, 39] and etc. All experiments are conducted from scratch without pre-training.

[1] https://github.com/Alibaba-MIIL/AudioClassfication

4.3. Result

Tab. 1 presents the results of ESC-50 and UrbanSound8K datasets. For ESC-50, we can observe that the proposed method demonstrates significant advantages. Compared to the original pooling module in EAT-S-AA, it can maximally compress the parameter size by approximately 98% (from $1969\,\mathrm{K}$ to $34.9\,\mathrm{K}$) and achieve slightly better performance ($91.25\,\%$ vs. $91.30\,\%$) with a $67.8\,\mathrm{K}$ model size. The overall model structure parameter size is also compressed by nearly $40\,\%$ (from 5.18M to 3.25M). The results of Speech Commands V2 presented in Tab. 2 again validate the improvement. In UrbanSound8k, due to the existence of a large number of silent pieces, the performance of all algorithms degrade as silent pieces do not contain any information, which also indicate that it is important to implement audio activity detection before classification. On this dataset, our methods can still achieve comparable performance with EAT-S-AA. We also experimented with different settings of GMME, and the results manifest that both kernel size N_k and the number of channels N_c affect accuracy. It is clear that a larger kernel size achieves better performance. One possible reason is that larger kernels can capture more local features. About channel widths, there is a similar tendency in channel numbers, this may be due to the enhancement of the anti-aliasing capability by the convolutional layer [30].

5. CONCLUSION

In this paper, we proposed an efficient pooling module for feature extraction, termed GMME, which integrates the advantages of attention. The proposed GMME can significantly reduce the number of parameters while preserving relevant information. GMME extracts time-domain and frequency-domain information through a mini-patch extraction. Then the interrelationship of the time-domain and frequency-domain information is modeled with the gate mechanism, a square kernel is carried out to extract global features. Compared to conventional anti-aliasing modules, GMME significantly reduces the model size. Experiments indicate that GMME could serve as a viable substitute for the process of audio classification.

6. REFERENCES

[1] M. Rajapakse and L. Wyse, "Generic audio classification using a hybrid model based on gmms and hmms," in *Proc. MMM*, 2005, pp. 53–58.

[2] I. Ikhsan, L. Novamizanti, and I. N. A. Ramatryana, "Automatic musical genre classification of audio using hidden markov model," in *Proc. ICoICT*, 2014, pp. 397–402.

[3] Y. Gong, Y.-A. Chung, and J. Glass, "PSLA: Improving audio tagging with pretraining, sampling, labeling, and Aggregation," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 3292–3306, 2021.

[4] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2880–2894, 2020.

[5] P. Gimeno, I. Viñals, A. Ortega, A. Miguel, and E. Lleida, "Multiclass audio segmentation based on recurrent neural networks for broadcast domain data," *EURASIP J.ASMP*, vol. 2020, no. 1, pp. 1–19, 2020.

[6] Q. Kong, Y. Xu, W. Wang, and M. D. Plumbley, "Sound event detection of weakly labelled data with cnn-transformer and automatic threshold optimization," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 28, pp. 2450–2460, 2020.

[7] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proc. Interspeech*, 2021, pp. 571–575.

[8] S. Chen, Y. Wu, C. Wang, S. Liu, and D. Tompkins, "BEATs: Audio pre-training with acoustic tokenizers," *arXiv preprint arXiv:2212.09058*, 2022.

[9] K. Chen, X. Du, B. Zhu, Z. Ma, T. Berg-Kirkpatrick, and S. Dubnov, "HTS-AT: A hierarchical token-semantic audio transformer for sound classification and detection," in *Proc. ICASSP*, 2022, pp. 646–650.

[10] J. Pons, O. Nieto, M. Prockup, E. Schmidt, A. Ehmann, and X. Serra, "End-to-end learning for music audio tagging at scale," in *Proc. ISMIR*, 2018.

[11] S. Verbitskiy, V. Berikov, and V. Vyshegorodtsev, "Eranns: Efficient residual audio neural networks for audio pattern recognition," *PRL*, vol. 161, pp. 38–44, 2022.

[12] M. Huzaifah, "Comparison of time-frequency representations for environmental sound classification using convolutional neural networks," *arXiv preprint arXiv:1706.07156*, 2017.

[13] I. Cohen, "Noise spectrum estimation in adverse environments: Improved minima controlled recursive averaging," *IEEE Trans. Speech Audio Process*, vol. 11, no. 5, pp. 466–475, 2003.

[14] H. Wang, D. Chong, D. Huang, and Y. Zou, "What affects the performance of convolutional neural networks for audio event classification," in *Proc. ACIIW*, 2019, pp. 140–146.

[15] X. Wang, J. Benesty, G. Huang, and J. Chen, "A minimum variance distortionless response spectral estimator with kronecker product filters," in *proc. EUSIPCO*, 2022, pp. 2261–2265.

[16] A. Gazneli, G. Zimerman, T. Ridnik, G. Sharir, and A. Noy, "End-to-end audio strikes back: Boosting augmentations towards an efficient audio classification network," *arXiv preprint arXiv:2204.11479*, 2022.

[17] R. Zhang, "Making convolutional networks shift-invariant again," in *Proc. ICML*, 2019, pp. 7324–7334.

[18] T. Tax, J. L. D. Antich, H. Purwins, and L. Maaløe, "Utilizing domain knowledge in end-to-end audio processing," in *Proc NIPS*, 2017.

[19] S. Venkatesh, D. Moffat, and E. R. Miranda, "You only hear once: a yolo-like algorithm for audio segmentation and sound event detection," *Applied Sciences*, vol. 12, p. 3293, 2022.

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[21] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proc. ICML*, 2019, pp. 6105–6114.

[22] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proc. ACM MM*, 2015, pp. 1015–1018.

[23] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[24] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *Proc. ACM MM*, 2014, pp. 1041–1044.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, and L. Jones, "Attention is all you need," *in Proc. NIPS*, vol. 30, pp. 5998—-6008, 2017.

[26] I. Tenney, D. Das, and E. Pavlick, "Bert rediscovers the classical nlp pipeline," in *Proc. ACL*, 2019, pp. 4593–4601.

[27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," *in Proc. ICLR*, 2021.

[28] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," in *Proc. ICLR*, 2021.

[29] G. Wu, Y. Liu, L. Fang, and T. Chai, "Revisiting light field rendering with deep anti-aliasing neural network," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 5430–5444, Sep 2022.

[30] A. H. Ribeiro and T. B. Schön, "How convolutional neural networks deal with aliasing," in *Proc. ICASSP*, 2021, pp. 2755–2759.

[31] L. Chi, B. Jiang, and Y. Mu, "Fast fourier convolution," *in Proc. NIPS*, vol. 33, pp. 4479–4488, 2020.

[32] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[33] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[34] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *in Proc. ICLR*, 2019.

[35] L. N. Smith, "A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay," *arXiv preprint arXiv:1803.09820*, 2018.

[36] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *in Proc. NIPS*, vol. 30, 2017.

[37] P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson, "Averaging weights leads to wider optima and better generalization," in *Proc. UAI*, 2018, pp. 876–885.

[38] Y. Tokozume and T. Harada, "Learning environmental sounds with end-to-end convolutional neural network," in *Proc. ICASSP*, 2017, pp. 2721–2725.

[39] S. Yun, D. Han, S. J. Oh, S. Chun, J. Choe, and Y. Yoo, "Cutmix: Regularization strategy to train strong classifiers with localizable features," in *Proc. ICCV*, 2019, pp. 6023–6032.