FlexiAST: Flexibility is What AST Needs

Jiu Feng*, Mehmet Hamza Erol*, Joon Son Chung, Arda Senocak

Korea Advanced Institute of Science and Technology, South Korea

{jiufeng2000,mehamerol,arda.senocak}@gmail.com, joonsc@kaist.ac.kr

Abstract

The objective of this work is to give patch-size flexibility to Audio Spectrogram Transformers (AST). Recent advancements in ASTs have shown superior performance in various audio-based tasks. However, the performance of standard ASTs degrades drastically when evaluated using different patch sizes from that used during training. As a result, AST models are typically re-trained to accommodate changes in patch sizes. To overcome this limitation, this paper proposes a training procedure to provide flexibility to standard AST models without architectural changes, allowing them to work with various patch sizes at the inference stage - FlexiAST. This proposed training approach simply utilizes random patch size selection and resizing of patch and positional embedding weights. Our experiments show that FlexiAST gives similar performance to standard AST models while maintaining its evaluation ability at various patch sizes on different datasets for audio classification tasks.

Index Terms: Transformer, Audio Spectrogram Transformers, Audio Classification

1. Introduction

The latest developments in transformer models [1] that rely purely on attention have had a significant impact in both audio processing [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13] and computer vision [14, 15, 16, 17, 18, 19, 20, 21, 22, 23] fields. When it comes to audio classification, all of the recent transformer-based approaches use a patch-based system. The input spectrograms are divided into fixed-size patches to create the tokens used for the transformer's inputs. While patches are an essential aspect of transformer models, prior approaches have primarily concentrated on the training paradigms for audio spectrogram transformers. The AST [2] adapts vision transformers to audio spectrograms and applies supervised learning with initialization from ImageNet [24] pre-trained ViT [14, 19] models, while SSAST [3] extends it with self-supervision. The complexity of the transformer models increases drastically with the input sequence length. Thus, efficient ways of reducing the complexity is crucial. MAE-AST [6] improves upon SSAST by utilizing a masked auto-encoder (MAE) approach to reduce the number of tokens. Additionally, several concurrent studies also explore some methods for optimizing the computational complexity of transformers through masking, patchout techniques and pooling, including [5, 7, 8, 9, 13].

Our main focus differs from other AST-based methods in that we prioritize the patch size of the audio spectrogram transformers (AST). While [2, 6] employ a fixed 16x16 patch size,
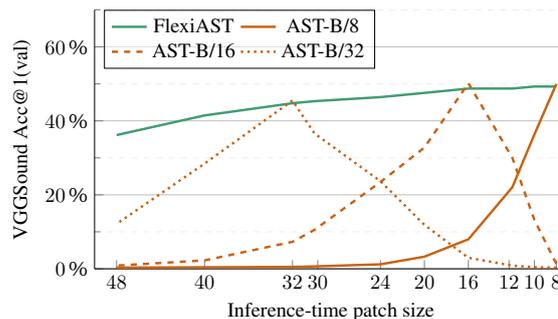

Figure 1: ***Standard ASTs vs. FlexiAST.*** *The performance of standard ASTs when evaluated on different patch sizes reveals their lack of flexibility.*

audio-based tasks vary in their requirements and may necessitate different patch sizes to achieve optimal performance. This typically entails re-training the AST (with certain patch size) to accommodate changes in patch sizes, since standard ASTs (including Vision Transformers - ViT [14]) function best only at their trained patch size. Adjusting an already trained AST with bilinear interpolation to evaluate on different patch sizes results in a significant performance degradation as Figure 1 displays. Thus, it is evident that standard ASTs lack flexibility.

This motivates us to design *one single AST model* that can seamlessly work at different patch sizes without significant performance degradation while achieving comparable performance to standard ASTs trained at fixed patch sizes. With this objective in mind, we explore a learning process that can give flexibility to a standard AST model. The closest work to ours is FlexiViT [25], which provides flexibility to Vision Transformers (ViT) to perform well at various patch sizes. We draw inspiration from the findings of this work and study the methodologies suitable for the flexibilization of standard ASTs. To accomplish this task, we take the following steps: (1) Instead of using a fixed patch size during training, we randomly select patch sizes (2) We resize the patch and positional embedding weights for the chosen patch size. The rest of the model (AST Transformer Encoder and Linear Layer) and training procedure remain identical to the standard AST, as depicted in Figure 2. Additionally, we explore how various tasks necessitate distinct resizing methodologies. For instance, audio classification tasks concentrate on audio context, and, as such, resizing patch embeddings in the frequency axis together with the time axis may not alter the underlying semantic information. Conversely, in speaker identification tasks, frequency is directly linked to speaker-specific information. Hence, we have explored that FlexiAST requires patch embedding resizing solely

along the time axis for speaker identification.

Our main contributions can be summarized as follows: (1) We demonstrate that standard ASTs are not flexible enough to be evaluated at patch sizes different from the size they were trained on; (2) We provide a training procedure that offers flexibility to standard ASTs without requiring architectural changes; (3) We show that this simple approach improves the flexibility of ASTs on different datasets, namely AudioSet [26], VGGSound [27], ESC-50 [28], Speech Commands [29], and VoxCeleb [30].

2. Approach

2.1. Preliminaries

Audio Spectrogram Transformer (AST) [2] is a transformer based architecture that takes an audio spectrogram $x \in \mathbb{R}^{f \times t}$ as its input. It tokenizes x into a sequence $x_i \in \mathbb{R}^{p \times p}$ of patches and computes the patch embeddings $e_i = (e_i^k)_{k=1}^d \in \mathbb{R}^d$ for all the patches through $e_i^k = vec(x_i)^T vec(\omega_k)$, where $\omega = (\omega_k)_{k=1}^d \in \mathbb{R}^{d \times p \times p}$ are the patch embedding weights, and vec function flattens a multi-dimensional array to a one-dimensional vector. Afterward, the trainable positional embeddings $\pi = (\pi_i)_{i=1}^{h \times w} \in \mathbb{R}^{(h \times w) \times d}$ are added to each of the patch embeddings $s_i = e_i + \pi_i$ for the final version of the tokens, where h and w represent the dimensions for the grid of tokens obtained. Finally, a [CLS] token is inserted at the beginning of the obtained token sequence, and the final sequence is fed into a multi-layered transformer encoder. The transformed output of the [CLS] token is used as the representation of the audio spectrogram for the downstream tasks. Since audio datasets typically do not contain large amounts of data required by transformers, AST initializes itself from a pretrained ViT weights to improve the training process and the performance.

2.2. ASTs are not Flexible

The patch embedding weights ω and positional embeddings π are the only components of an AST with parameterizations depending on the patch size. Thus, in case a pretrained AST model is evaluated at a different patch size than the original parameterization (trained fixed patch size), these two components must be adjusted appropriately. More specifically, if we want to evaluate a pretrained standard AST with patch embedding weights $\omega \in \mathbb{R}^{d \times p \times p}$ and positional embeddings $\pi \in \mathbb{R}^{(h \times w) \times d}$ at the patch size $2p \times 2p$, then the patch embedding weights and positional embeddings should be resized into $\hat{\omega} \in \mathbb{R}^{d \times 2p \times 2p}$ and $\hat{\pi} \in \mathbb{R}^{\lceil h/2 \rceil \times \lceil w/2 \rceil \times d}$ respectively (assuming no overlap of patches). To achieve this in a most conventional and intuitive way, one can simply use bilinear interpolation operation as the original ViT paper [14] use this resizing method on the positional embeddings. However, Figure 1 illustrates that the performance of the AST models collapses when evaluated at different patch sizes with this resizing methodology. This implies that simply resizing the patch embedding weights of an AST in an intuitive way does not yield flexibility across multiple patch sizes. In the following sections, we propose a methodology to train flexible AST models that perform well across multiple patch sizes without a significant performance drop.

2.3. Approach and Training

To achieve flexification, we only refine the training pipeline of standard ASTs in the light of FlexiViT's [25] findings, without making any architectural changes on standard ASTs. The new training pipeline involves two key stages: (1) selecting
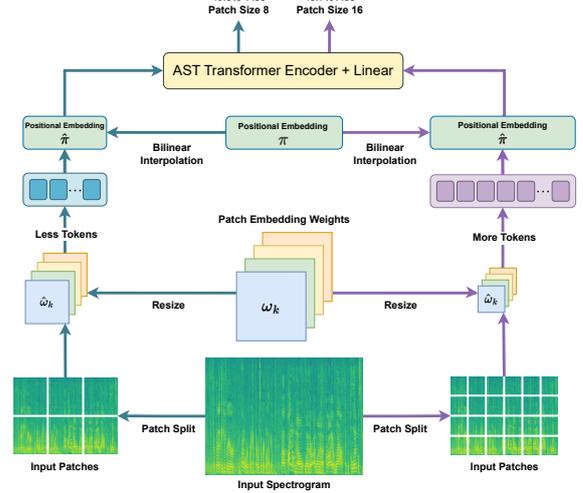

Figure 2: **Our FlexiAST framework.** *It is a standard AST model (architecturally identical) with random patch size selection and resizing operation on patch embeddings weights during training.*

a random patch size during training as opposed to a fixed patch size, and (2) adjusting the weights of the patch embeddings based on the chosen patch size. We randomly select a patch size $\hat{p}$ during training from a set of patch sizes $P = \{8, 10, 12, 16, 20, 24, 30, 32, 40, 48\}$. Then, we resize the patch embedding weights and positional embeddings appropriately to match the selected patch size. The remaining training process is the same as standard ASTs, shown in Figure 2.

While bilinear interpolation can be used to resize the positional embeddings for a randomly selected patch size, resizing the patch embedding weights require more careful design choices. We mainly investigate two approaches in this work, namely bilinear interpolation (BL) and pseudo-inverse resize (PI-resize), which is inspired from FlexiViT. For a patch at different resolutions obtained through bilinear interpolation, the PI-Resize operator is designed to adjust the patch embedding weights to maximize the alignment of the extracted information. More formally, resizing a patch x of size $p \times p$ into $\hat{x}$ of size $\hat{p} \times \hat{p}$ through bilinear interpolation could be represented as a linear transformation:

$$\hat{x} = B_p^{\hat{p}} vec(x) \tag{1}$$

where $B_p^{\hat{p}} \in \mathbb{R}^{\hat{p}^2 \times p^2}$ is the bilinear interpolation matrix. Then, PI-Resize operator aims to obtain resized patch embedding weights $\hat{\omega}$ that satisfy the following optimization objective:

$$\hat{\omega} \in \arg\min_{\hat{\omega}} \mathbb{E}_{x \sim \mathcal{X}} \left[(\langle x, \omega \rangle - \langle \hat{x}, \hat{\omega} \rangle)^2 \right] \tag{2}$$

where $\langle a, b \rangle = vec(a)^T vec(b)$ and $\mathcal{X}$ is a distribution over the patches. For both of the upsampling ($\hat{p} \geq p$) and the downsampling ($\hat{p} < p$) cases, the $\hat{\omega}$ is recovered as:

$$\hat{\omega} = P_p^{\hat{p}} vec(\omega) \tag{3}$$

where $P_p^{\hat{p}} \in \mathbb{R}^{\hat{p}^2 \times p^2}$ is the pseudoinverse of $(B_p^{\hat{p}})^T$, thus the matrix representing the PI-Resize transformation. Although PI-Resize and resizing the spectrogram are related in function, they differ in that PI-Resize modifies the model parameters rather

than the input spectrogram [25]. FlexiAST follows the same architecture and training process as standard ASTs after refining the weights of patch embeddings and positional embeddings for randomly selected patch sizes. In this work, supervised learning and PI-resize functions are employed as the default training and patch embedding weights resizing approaches respectively. These design choices are validated in Section 3.4.

3. Experiments

3.1. Datasets and Evaluation Metrics

Datasets. In our experiments, we train our method on (1) AudioSet Full, (2) VGGSound, (3) ESC-50, (4) Speech Commands, and (5) VoxCeleb datasets. **AudioSet** [26] is a large-scale dataset that contains a diverse collection of audio samples, each labeled with a set of annotations. The dataset consists of over 2 million 10-second clips, covering a wide range of categories such as musical instruments, animal sounds, and human speech. There are a total of 527 labels. **VGGSound** [27] is a dataset that consists of nearly 200K 10-second videos. Each clip has been labeled with 309 sound classes, which include objects, human actions, and human-object interactions. **ESC-50** [28] dataset comprises 2,000 environmental audio recordings, each lasting for 5 seconds and classified into 50 categories. **Speech Commands-V2** [29] contains ~105K recordings, each lasting for 1 second, and featuring 35 commonly used speech commands. **VoxCeleb** [30] is an audio-visual human speech dataset with 1251 speakers and ~145K utterances.

Evaluation metrics. Due to the presence of multiple labels in each sample in AudioSet, we evaluate it using the mean average precision (mAP) across all classes. In contrast to AudioSet, we report the Top-1 classification accuracy for the remaining datasets since only a single label is assigned for each sample.

3.2. Implementation Details

Details of FlexiAST. FlexiAST is architecturally identical to AST, with only differences in random patch size selection and patch resizing steps during training. Therefore, it has the same underlying patch size parameters as AST, which is set to 16x16 (ω for FlexiAST). The original AST provides additional techniques to further improve performance, such as patch overlapping, mixup augmentation [31], padding, and model aggregation [32, 33]. For simplicity and to observe the direct impact of flexification, we do not use the above-mentioned settings in FlexiAST or standard ASTs presented in this paper. The rest of the default settings of AST are adapted identically. The original AST demonstrates that initialization enhances performance, and we also employ initialization in FlexiAST. However, unlike standard AST, FlexiAST obtains initialization weights from standard ASTs such as AST-B/8, AST-B/16, or AST-B/32 instead of ViT [14] or DeiT [19]. When initializing the weights of the FlexiAST, we adjust the patch embedding weights and positional embeddings of standard AST with PI-resize and bilinear interpolation respectively. In addition, we choose Supervised Training and PI-resize as the default options for the training paradigm and patch resizing operation, respectively, rather than Knowledge Distillation (KD) and bilinear interpolation (BL). The validation of these choices is discussed in Section 3.4.

Details of Standard ASTs. Unless specified otherwise, all standard ASTs (B/8, B/16, and B/32) within a particular dataset are trained for the same number of epochs, initialized with ViT, and learned using supervised learning with fixed patch sizes. However, for ESC-50 and Speech Commands ASTs, we ini-

Model	48	40	32	30	24	20	16	12	10	8
VGGSound (Acc)										
AST-B/8	0.003	0.004	0.005	0.006	0.012	0.033	0.080	0.221	0.363	0.500
AST-B/16	0.009	0.023	0.073	0.105	0.234	0.328	0.502	0.300	0.132	0.013
AST-B/32	0.121	0.284	0.455	0.366	0.237	0.119	0.030	0.008	0.005	0.003
FlexiAST-KD	0.362	0.415	0.448	0.453	0.464	0.476	0.487	0.487	0.493	0.493
FlexiAST-Sup.	0.348	0.400	0.432	0.438	0.456	0.467	0.481	0.480	0.478	0.484
AudioSet (mAP)										
AST-B/8	0.006	0.006	0.006	0.006	0.007	0.009	0.025	0.151	0.276	0.397
AST-B/16	0.008	0.011	0.028	0.044	0.172	0.297	0.396	0.248	0.100	0.024
AST-B/32	0.074	0.211	0.371	0.348	0.181	0.069	0.021	0.010	0.007	0.006
FlexiAST-KD	0.306	0.337	0.353	0.362	0.378	0.387	0.394	0.399	0.397	0.397
FlexiAST-Sup.	0.305	0.338	0.355	0.365	0.380	0.390	0.396	0.401	0.400	0.399
ESC-50 (Acc)										
AST-B/8	0.020	0.018	0.022	0.028	0.048	0.060	0.107	0.517	0.765	0.950
AST-B/16	0.045	0.070	0.107	0.163	0.463	0.765	0.948	0.487	0.253	0.095
AST-B/32	0.310	0.623	0.943	0.895	0.465	0.195	0.070	0.020	0.015	0.035
FlexiAST-KD	0.780	0.848	0.877	0.897	0.895	0.912	0.927	0.932	0.927	0.940
FlexiAST-Sup.	0.752	0.870	0.882	0.892	0.910	0.920	0.930	0.943	0.930	0.940
Speech Commands (Acc)										
AST-B/8	0.029	0.037	0.045	0.045	0.062	0.081	0.185	0.682	0.884	0.960
AST-B/16	0.051	0.083	0.252	0.314	0.734	0.882	0.973	0.864	0.681	0.382
AST-B/32	0.230	0.616	0.967	0.944	0.623	0.484	0.148	0.041	0.025	0.021
FlexiAST-KD	0.826	0.875	0.896	0.897	0.908	0.915	0.923	0.923	0.927	0.926
FlexiAST-Sup.	0.823	0.893	0.921	0.929	0.944	0.951	0.960	0.962	0.964	0.964

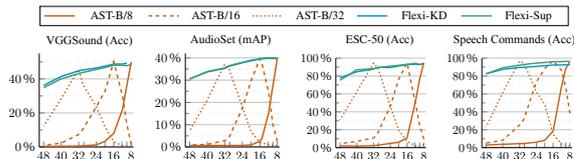Table 1: *Comparison of Standard ASTs and FlexiAST.*

Figure 3: *Visualization of Table 1.*

tialize them from AudioSet pre-trained standard ASTs and then fine-tune them. After obtaining each standard AST in various datasets, AST-B/8 are used to initialize the FlexiAST. However, only for VoxCeleb dataset, AST-B/16 is employed.

3.3. Results of Flexification

This section presents a comparison between our FlexiAST and the conventional ASTs trained using fixed patch sizes. We assess the performance of all models in inference stage across multiple patch sizes on five datasets, namely AudioSet, VGGSound, ESC-50, Speech Commands, and VoxCeleb. The evaluation results for first four dataset are presented in Table 1 and Figure 3, and VoxCeleb results are displayed in Figure 4.

Results for AudioSet, VGGSound, ESC-50 and Speech Commands. Our results demonstrate that our FlexiAST displays a high degree of flexibility to handle a broad range of patch sizes during inference on all datasets, regardless of the training paradigm, whether it is *Supervised* or *Knowledge Distillation*. In contrast, the performance of standard ASTs deteriorates when evaluated on patch sizes different from those used during training. These results confirm that standard ASTs can be flexified with the proposed approach without any architectural changes. Notably, our FlexiAST model also performs comparably to the standard ASTs (or outperforms in
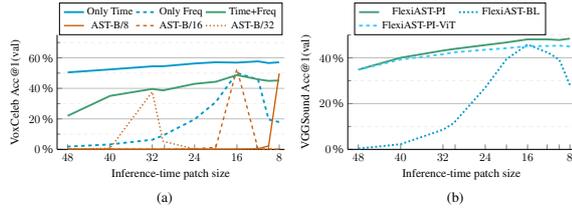
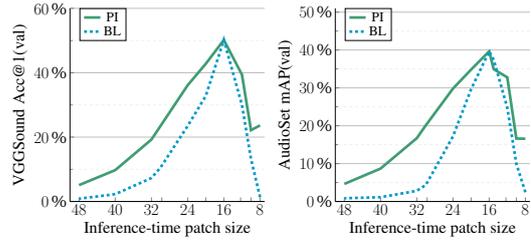Figure 4: *FlexiAST design choices on VoxCeleb and VG-GSound.*

Figure 5: *The effect of PI-resize in standard ASTs. We evaluate the performance of Standard ASTs on various patch sizes with PI-resize and bilinear interpolation to see the impact of each resizing operation.*

some cases) when tested on the specific patch size they were trained on, such as FlexiAST-Sup. gives higher performance than AST-B/8 when evaluated at patch size 8 on AudioSet (Refer to Table 1). This concludes that our exploration provides a single model that can perform as good as a standard AST while retaining its ability to perform well when evaluated on various patch sizes.

Results for VoxCeleb. We find that when we apply our default flexification procedure (random patch size selection during training and resizing operation on patch embeddings), FlexiAST does not perform well on VoxCeleb dataset. The classification task in above mentioned datasets involves detecting the context of the samples, and so the resizing operation may not have a big impact since the context remains similar. However, the task of speaker identification in the VoxCeleb dataset requires speaker-specific information, in which the context of the sample is irrelevant, i.e., speaker embedding. Therefore, resizing patches in the frequency axis can directly affect the speaker-related information. To overcome this problem, we resize the patch size only in the time axis and apply resizing for the patch embedding weights accordingly during training. The result of this procedure is shown in Figure 4 (a), demonstrating that this simple modification enables FlexiAST to work as expected and even perform beyond the standard ASTs. To further support this, we train FlexiAST by solely applying patch embedding resizing in the frequency axis. Results in Figure 4 (a) demonstrate that resizing along the frequency axis has a significant harm on the flexibility of ASTs in speaker identification tasks.

3.4. Ablation Study

We perform a series of ablative experiments to demonstrate the effectiveness of the techniques used in our proposed model and their alternatives. In order to reduce computational resources and time requirements, we mainly conduct the experiments on the VGGSound dataset. Key findings are as follows:

1. Both supervised learning and knowledge distillation give good flexibility as a training paradigm. We compare the performance of FlexiAST with different training paradigms. Specifically, we provide results in two settings: (1) Training with supervised learning, (2) Training with knowledge distillation (KD). For training with knowledge distillation, we use KL divergence as our loss and distill from the same standard AST model being used to initialize the FlexiAST model. The performance of these approaches is presented in Table 1. As depicted in the results, both training methodologies give similar performance generally and show marginal differences depending on the datasets, such as the Speech Commands dataset, where supervised learning shows a modest gap. However, due to its simplicity and shorter training time, we select the supervised learning as our default training method.

2. PI-resize method provides flexibility while bilinear interpolation fails. To demonstrate the impact of different resizing

methods on FlexiAST, we utilize two distinct techniques, bilinear interpolation and PI-resize. In this experiment, we use FlexiASTs trained on VGGSound with supervised learning. The results of the experiment are presented in Figure 4 (b), where we compare the effectiveness of the two methods. It is observed that while PI-resize makes successful flexification, bilinear interpolation (BL) fails to provide the desired flexibility.

3. Utilizing PI-resize as a resizing technique to evaluate standard ASTs in various patch sizes helps for relatively better performance. The results of the previous experiment highlight that PI-resize is capable of providing flexibility to FlexiAST, whereas bilinear interpolation fails to do so. Additionally, standard ASTs are not flexible when they are evaluated on different patch sizes with bilinear interpolation (BL) as a conventional method. This leads us to question whether the use of PI-resize can improve the evaluation performance of standard ASTs on various patch sizes. Figure 5 presents comparison results obtained when PI-resize and BL are employed. Our findings indicate that standard ASTs do not perform well on patch sizes other than those they are trained on, regardless of the resizing method is used. However, PI-resize outperforms bilinear interpolation in terms of performance.

4. Initialization from standard AST leads to a performance improvement. We examine the impact of the source of initialization on FlexiAST in our experiment. We conduct two sets of experiments: (1) initializing FlexiAST from a standard AST trained on the same dataset using an 8x8 patch size, and (2) initializing FlexiAST from standard ViT. Figure 4 (b) illustrates the experiment results that FlexiAST with standard AST initialization performs better than ViT initialization on the VGGSound dataset. However, It should be noted that even with the ViT initialization, our proposed method can still flexify.

4. Conclusion

In this paper, we focus on providing patch-size flexibility to ASTs. We identify that standard ASTs, which are trained with fixed patch sizes, lack flexibility, resulting in significant performance degradation when evaluated on various patch sizes that differ from the sizes they were trained on. To address this issue, we introduce a training approach that offers flexibility to existing ASTs without requiring any architectural changes. This is achieved by simply utilizing random patch size selection and resizing patch embedding weights accordingly. Additionally, we explore the necessity of task-specific resizing techniques, such as only resizing the time axis in speaker identification, to achieve optimal flexibility. As a result, our approach enables the creation of FlexiAST, a model that is versatile enough to handle all patch sizes with ease.

5. References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[2] Y. Gong, Y.-A. Chung, and J. Glass, "AST: audio spectrogram transformer," in *Proc. Interspeech*, 2021.

[3] Y. Gong, C.-I. Lai, Y.-A. Chung, and J. Glass, "SSAST: self-supervised audio spectrogram transformer," in *Proc. AAAI*, 2022.

[4] 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.

[5] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," in *Proc. Interspeech*, 2022.

[6] A. Baade, P. Peng, and D. Harwath, "MAE-AST: masked autoencoding audio spectrogram transformer," in *Proc. Interspeech*, 2022.

[7] P.-Y. Huang, H. Xu, J. B. Li, A. Baevski, M. Auli, W. Galuba, F. Metze, and C. Feichtenhofer, "Masked autoencoders that listen," in *NeurIPS*, 2022.

[8] D. Chong, H. Wang, P. Zhou, and Q. Zeng, "Masked spectrogram prediction for self-supervised audio pre-training," *arXiv preprint arXiv:2204.12768*, 2022.

[9] D. Niizumi, D. Takeuchi, Y. Ohishi, N. Harada, and K. Kashino, "Masked spectrogram modeling using masked autoencoders for learning general-purpose audio representation," in *PMLR*, 2022.

[10] A. Nagrani, S. Yang, A. Arnab, A. Jansen, C. Schmid, and C. Sun, "Attention bottlenecks for multimodal fusion," in *NeurIPS*, 2021.

[11] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *TASLP*, 2021.

[12] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *NeurIPS*, 2020.

[13] W. Zhu and M. Omar, "Multiscale audio spectrogram transformer for efficient audio classification," in *Proc. ICASSP*, 2023.

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

[15] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," in *Proc. CVPR*, 2019.

[16] J. Lu, D. Batra, D. Parikh, and S. Lee, "ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," in *NeurIPS*, 2019.

[17] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Proc. ECCV*, 2020.

[18] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proc. ICCV*, 2021.

[19] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *Proc. ICML*, 2021.

[20] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr *et al.*, "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers," in *Proc. CVPR*, 2021.

[21] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," in *Proc. ICCV*, 2021.

[22] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *Proc. ICCV*, 2021.

[23] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, and C. Feichtenhofer, "MViTv2: Improved multiscale vision transformers for classification and detection," in *Proc. CVPR*, 2022.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, 2009.

[25] L. Beyer, P. Izmailov, A. Kolesnikov, M. Caron, S. Kornblith, X. Zhai, M. Minderer, M. Tschannen, I. Alabdulmohsin, and F. Pavetic, "FlexiViT: one model for all patch sizes," *arXiv preprint arXiv:2212.08013*, 2022.

[26] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "AudioSet: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*, 2017.

[27] H. Chen, W. Xie, A. Vedaldi, and A. Zisserman, "VGGSound: A large-scale audio-visual dataset," in *Proc. ICASSP*, 2020.

[28] K. J. Piczak, "ESC: Dataset for environmental sound classification," in *Proc. ACMM*, 2015.

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

[30] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "VoxCeleb: Large-scale speaker verification in the wild," *Computer Speech & Language*, 2020.

[31] Y. Tokozume, Y. Ushiku, and T. Harada, "Learning from between-class examples for deep sound recognition," in *Proc. ICLR*, 2018.

[32] 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.

[33] L. Breiman, "Bagging predictors," *Machine Learning*, 1996.