Audio Mamba: Bidirectional State Space Model for Audio Representation Learning

Anonymous Submission

Abstract

Transformers have rapidly become the preferred choice for audio classification, surpassing methods based on CNNs. However, Audio Spectrogram Transformers (ASTs) exhibit quadratic scaling due to self-attention. The removal of this quadratic self-attention cost presents an appealing direction. Recently, state space models (SSMs), such as Mamba, have demonstrated potential in language and vision tasks in this regard. In this study, we explore whether reliance on self-attention is necessary for audio classification tasks. By introducing Audio Mamba (AuM), the first self-attention-free, purely SSM-based model for audio classification, we aim to address this question. We evaluate AuM on various audio datasets - comprising six different benchmarks - where it achieves comparable or better performance compared to well-established AST model.

1. Introduction

In recent years, CNNs [15, 18] have been replaced with transformer-based architectures [7, 10, 26, 27] in a paradigm shift in deep learning, as transformers outperform convolutional neural networks. Not only does the performance of transformers exceed that of CNNs, but establishing a unified architecture among many different research fields and tasks — traditionally using completely different models — is another breakthrough [1, 2, 11, 12, 21, 23, 24, 29, 30]. Despite their success, transformers are hindered by their reliance on the computationally intensive self-attention mechanism. The $\mathcal{O}(n^2)$ cost of attention is a natural concern when processing longer sequences. This limitation motivates the exploration of alternative architectures, notably state-space models (SSMs) [8, 13, 14, 25] such as Mamba [13], which replaces the self-attention mechanism in favor of incorporating time-varying parameters to capture global context efficiently. Recently, the introduction of Mamba [13] marks a significant advancement in model efficiency for both training and inference, suggesting a potential alternative to transformer-based approaches. Given the universality and scalability of transformers across various tasks, Mamba's potential, coupled with its computational efficiency, is particularly promising for becoming a similarly generic and versatile architecture.

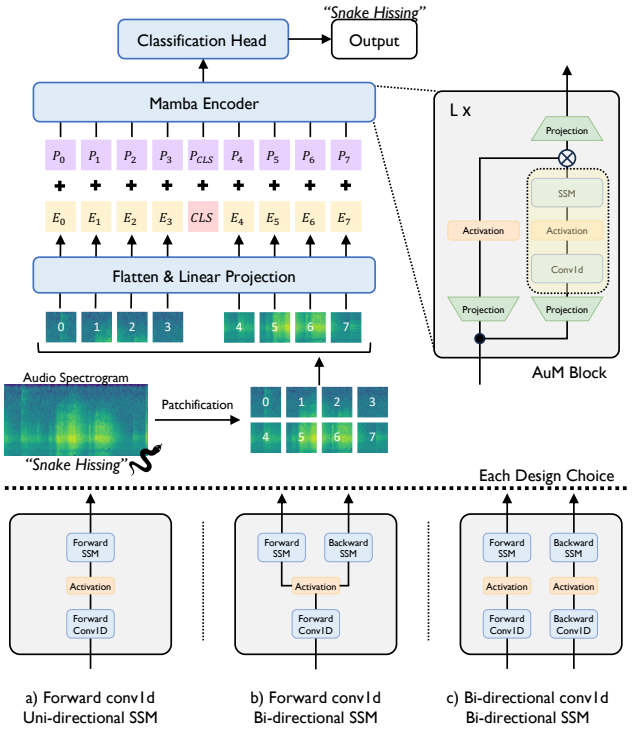

Figure 1. **The proposed Audio Mamba (AuM) architecture.**

Despite Mamba's recent successes in language modeling and vision [19, 20, 31–33], the adoption of Mamba and similar SSM-based models in the audio classification domain still remains unexplored. This gap motivates our work, where we introduce a novel SSM-based model, Audio Mamba - AuM, applied directly to audio spectrograms. Our approach is self-attention free, focusing purely on long sequence modeling with state space models. AuM not only achieves comparable performance to the Audio Spectrogram Transformer (AST) [10], the most prominent approach in audio classification, but also retains several advantages of transformer-based models. These include the ability to handle varying sequence lengths and the ease of transferability to other tasks. Due to the employment of

state space models, reliance on self-attention is eliminated, enabling the model to operate with linear time complexity relative to sequence length and feature dimension, as opposed to AST where quadratic complexity is observed. The closest work to ours is Vision Mamba [33], which utilizes bidirectional SSM for global visual context modeling and positional embeddings for location information in a structure similar to Vision Transformers (ViT) [7]. Drawing on AST's success in applying ViT's principles to audio classification, we also draw inspiration from the findings of Vision Mamba and study the methodologies suitable for applying bidirectional state space models to audio classification. To accomplish this task, we take the following steps: (1) We divide the input spectrogram into patches, which are then projected into patch embedding tokens. (2) We add an additional learnable classification token to the sequence of patch tokens, specifically in the middle. (3) The Audio Mamba Encoder blocks process these token sequences in both forward and backward directions with SSM modules. (4) The classification token is utilized to train the model on the supervised audio classification task and also for making predictions in the inference stage. We summarize the contributions of our work as follows:

- We introduce Audio Mamba (AuM) for processing audio spectrograms, utilizing bidirectional state space models (SSM) to handle tokens in both forward and backward directions, in a similar structure of Audio Spectrogram Transformers (AST).
- By eliminating self-attention modules, AuM achieves linearly scaled resource consumption when evaluated with long audio sequences.
- Our comprehensive experiments across six diverse datasets — AudioSet [9], AudioSet Balanced, VGGSound [4], VoxCeleb [22], Speech Commands V2 [28], and Epic-Sounds [17] — show that AuM delivers performance that is comparable to or exceeds the most prominent audio classification method AST.

2. Audio Mamba

2.1. Flow of the Architecture

The Audio Mamba (AuM) architecture, as depicted in Fig. 1, begins by transforming an input audio waveform into an audio spectrogram $X \in \mathbb{R}^{F \times T}$, where F and T represent the frequency and time dimensions, respectively. The spectrogram is partitioned into a sequence of N square patches $S \in \mathbb{R}^{N \times p \times p}$, with p denoting the side length of each patch and N calculated as $N = (F/p) \times (T/p)$. Each individual patch S_i is subsequently flattened into a one-dimensional vector $S_i \in \mathbb{R}^{p^2}$, and through a linear projection, it is embedded into a D-dimensional space, yielding $E_i \in \mathbb{R}^D$. This process is facilitated by the patch embedding layer. Afterward, a special learnable classification token, denoted as $CLS \in \mathbb{R}^D$, is inserted into the middle of the sequence, leading to an augmented embedding sequence $E \in \mathbb{R}^{(N+1) \times D}$. To encode the position of each element within the sequence, learnable positional embeddings $P \in \mathbb{R}^{(N+1) \times D}$ are added, resulting in the token sequence $T \in \mathbb{R}^{(N+1) \times D}$. This token sequence is then processed by the Audio Mamba encoder which consists of L stacked blocks, each of which retains the dimensionality of its input. Thus, the encoder transforms T into an output sequence $T' \in \mathbb{R}^{(N+1) \times D}$. The modified representation of the classification token $T'_{N/2}$ is then conveyed to the classification head.

2.2. Architecture Details

Aiming to establish itself as a generic architecture, AuM shares several similarities with the AST [10]. However, AuM distinguishes itself through distinct components and strategic design decisions that highlight its unique architectural and operational characteristics, to be a self-attention-free model.

Preliminaries. State space models (SSMs) are linear time-invariant systems that aim to model a continuous system which maps a time dependent D dimensional input sequence $x(t) \in \mathbb{R}$ to an output $y(t) \in \mathbb{R}$ through maintaining a hidden state $h(t) \in \mathbb{R}^N$. Such a system could be represented with the following equation:

$$\begin{aligned} h'(t) &= Ah(t) + Bx(t), \\ y(t) &= Ch(t). \end{aligned} \tag{1}$$

where $A \in \mathbb{R}^{N \times N}$, $B \in \mathbb{R}^{N \times D}$ and $C \in \mathbb{R}^{D \times N}$. With the primary goal of adapting the model to deep learning algorithms, a discretization process is applied, which transforms the continuous parameters A and B through a discretization rule into $\bar{A}$ and $\bar{B}$, respectively. These discretized parameters are then substituted for A and B, leading to the following discretized formulation of the system:

$$\begin{aligned} h_t &= \bar{A}h_{t-1} + \bar{B}x_t, \\ y_t &= Ch_t. \end{aligned} \tag{2}$$

Such a linear time-invariant system could be computed both as a linear recurrence or through a global convolution, enabling efficient processing [13]. Despite its efficiency, such a system has limitations in modeling certain types of data due to its time-invariant parameterization. Mamba upgrades existing works based on such models by converting the time-invariant parameters into a time-variant format, enabling efficient derivation of parameters from time-varying inputs. Specifically, inside the Forward SSM module of a Mamba block (Fig. 1 (a)), the algorithm utilizes the input sequence $x \in \mathbb{R}^{L \times D}$ that has been convolved through a Forward Conv1D before, to convert each time-invariant parameter A, B, and Δ in eq. (2) into specific corresponding

ones A'_i, B'_i, and Δ'_i for each element x_i of the sequence. Mamba then utilizes these parameters by adopting a modern hardware-oriented scanning method that processes the input sequence from beginning to end in a unidirectional manner. More details can be seen in [13]. This enables the model to selectively update its hidden state by capturing relevant information from the input sequence through these converted parameters.

Bidirectional Mamba Encoder. Even though Mamba's unidirectional scan of the sequence offers promising benefits for modeling causal sequential data, its application to learning from 2D data benefits from processing in multiple directions [19, 32, 33]. For instance, in learning from visual data, an existing Mamba-based architecture, ViM [33], modifying the original Mamba block in Fig. 1 (a) to Fig. 1 (c) by introducing another direction for feature extraction (Backward Conv1D) or scanning (Backward SSM) of the input sequence, enabling multi-directional and spatial-aware processing. Similarly, AuM adopts the design strategy shown in Fig. 1 (b) by adding an extra backward scanning direction to the original Mamba block. This approach utilizes the same convolved features while adapting both the forward and backward SSM parameters into their time-variant (input-dependent) versions for scanning. Likewise ViM, this enables AuM to model the global context in a spatial aware manner, mirroring the functionality of self-attention mechanism in transformers for modeling global context.

Classification Token. Unlike transformers in their pure form, which are permutation invariant when processing the input sequence, AuM block is sensitive to the order of the input sequence because both feature extraction (Conv1D) and SSMs are input-order-sensitive operations. Consequently, in addition to scanning directions, the placement of the classification token within the input sequence becomes critical for the learning process. Similarly to ViM, AuM strategically positions the classification token at the midpoint of the input sequence, immediately after the patch embedding layer. This setup has shown improved performance in bidirectional processing setup, as demonstrated in ViM, and the ablation study conducted in Section 3.

3. Experiments

3.1. Datasets and Evaluation Metrics

Datasets. Our experiments utilize: (1) Audioset Full / Balanced, (2) VGGSound, (3) VoxCeleb, (4) Speech Commands-V2, and (5) EPIC-SOUNDS datasets. **AudioSet** [9] is an expansive dataset with a wide array of audio samples, each marked with a set of labels. It includes over 2 million 10 seconds long audio clips with a total of 527 distinct labels. The balanced set on the other hand is curated from the full set, consisting of 20K samples. **VGGSound** [4] contains nearly 200k video clips of 10 seconds each, annotated with 309 diverse sound categories. **VoxCeleb** [22] is a dataset focused on audio-visual representations of human speech, featuring 1,251 speakers and around 145k speech instances. **Speech Commands-V2** [28] comprises approximately 105k audio recordings, each with a duration of 1 second, and includes 35 widely recognized speech commands. Finally, **EPIC-SOUNDS** [17], part of EPIC-KITCHENS-100 [6], comprises 75.9k audio segments from egocentric videos, labeled across 44 classes, focusing on actions discernible by sound, such as object collisions with material annotations.

Evaluation metrics. We utilize mean average precision (mAP) for Audioset experiments due to the existence of multiple labels per sample. For the remaining datasets, we show the top-1 classification accuracy (Acc) as the samples have a single label.

3.2. Comparison to AST on Standard Benchmarks

In this section, we conduct a comparative analysis of our Audio Mamba (AuM) against the Audio Spectrogram Transformer (AST) model. Both models use base backbones, AuM-B/16 and AST-B/16. As discussed in the implementation details, we follow the same training and experimental settings as the AST model to ensure a fair comparison, which are detailed at section 5. It is worth highlighting that in this experiment, neither AuM nor AST utilized pretraining weights from other models (AST is initialized with weights from the Vision Transformer (ViT) model pretrained on ImageNet in the original paper [10]) to ensure a pure comparison of these two different architectures. We repeat each experiment three times with the same setup but different random seeds and report the results with the mean and standard deviation in Table 1. Our proposed *AuM generally achieves better performance* in this experimental setup. This indicates that AuM, with its pure setting, is a potential alternative to the AST model, without relying on self-attention, which leads to better efficiency in computational resources.

3.3. Comparison to AST on Efficiency

Transformer-based audio classification models are computationally demanding (quadratic complexity), particularly with lengthy audio and high-dimensional data. SSM-based models stand out for their computational and memory efficiency. In this section, we compare AuM to AST from an efficiency perspective. A single A6000 GPU is used for this experiment. We feed audios with corresponding lengths for every given token number to the models to simulate the speed and GPU memory comparison. We visualize the speed and memory consumption of these models in Figure 2. AuM demonstrates clear computation and memory efficiency. For example, the AuM-Base model that uses 20

Model	AudioSet (mAP)	AS-20K (mAP)	VGGSound (Acc.)	VoxCeleb (Acc.)	Speech Comm. V2 (Acc.)	Epic-Sounds (Acc.)
AST-B/16	29.10 ± 0.07	10.41 ± 0.32	37.25 ± 0.31	22.44 ± 0.19	85.27 ± 1.07	**44.76** ± 0.20
AuM-B/16	**32.43** ± 0.31 (+3.33)	**13.28** ± 1.07 (+2.87)	**42.58** ± 0.28 (+5.33)	**28.34** ± 3.38 (+5.90)	**91.58** ± 3.17 (+6.32)	44.17 ± 0.58 (-0.60)

Table 1. **Results of from-scratch training of AST and AuM base models across various datasets.**

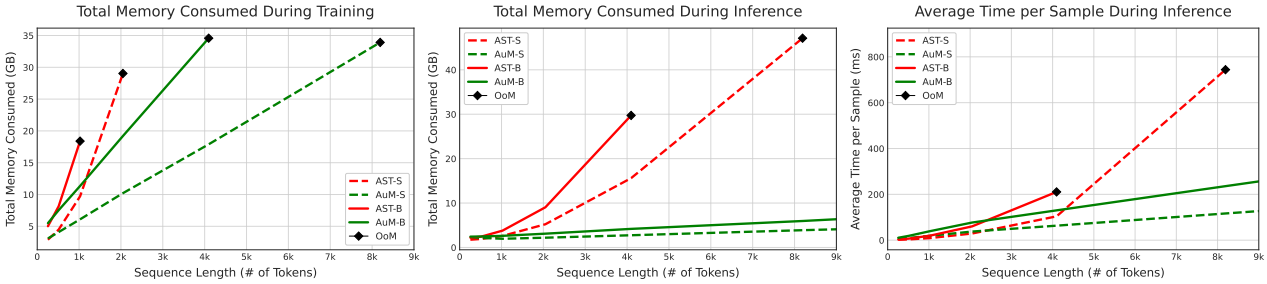

Figure 2. **Empirical evaluation of memory and time consumption for AST and AuM small/base models.**

Setting	AS-20K (mAP)			VGGSound (Acc.)		
	Head	Mid	End	Head	Mid	End
Fo-Fo (a)	0.48± 0.00	11.73± 0.47	11.90± 1.05	0.33± 0.00	35.05± 0.80	39.09± 0.56
Fo-Bi (b)	13.57± 0.09	**13.81**± 0.32	12.35± 0.33	40.91± 0.47	**42.58**± 0.28	40.41± 0.15
Bi-Bi (c)	4.97± 0.13	9.69± 0.43	11.11± 0.66	34.22± 0.26	36.48± 0.46	41.09± 0.16

Table 2. **Results of ablation on the design choices.** Architectural choices (under the settings column) refer to the block types in Fig. 1, the location of the classification token is indicated through the columns: Head, Mid and End per dataset.

seconds of audio (1024 tokens) for training consumes as little GPU memory as the AST-Small model. Additionally, while AuM-B can be trained with 80 seconds, AST-B will run out of memory in the setting that uses only 20 seconds audios. Moreover, AuM is 1.6 times faster in the inference stage than AST at 4096 number of tokens, with a growing rate as the token count increases. All these results indicate that AuM exhibits a trend of linear scaling with respect to sequence length.

3.4. Ablation Study on Design Choices

We conduct a series of experiments to verify our design choices and perform further analysis. We study the following strategies in terms of the direction of SSM modules and conv1Ds:

AuM-ForwardConv1D-**Fo**rwardSSM: This choice, which is the default Mamba block, directly applies the AuM Block with only a forward SSM (refer to Figure 1 (a)).

AuM-ForwardConv1D-**Bi**DirectionalSSM: This is the design of our final model, which applies an additional backward SSM to the previous design choice (refer to Figure 1 (b)).

AuM-Bi DirectionalConv1D-**Bi**DirectionalSSM: In this variant, we add another Conv1D in the backward direction to feed the output of this module to the backward SSM, making each SSM module a separate stream. A similar design is adopted in Vision Mamba (ViM) as the default choice (refer to Figure 1 (c)).

Moreover, the position of class tokens is ablated for each variant above. To save computational time and resources, we primarily conduct ablation studies by training our model on AudioSet Balanced (AS-20K) and VGGSound. Results are in Table 2.

Impact of bidirectional SSM. To understand the impact of the directions of SSM modules, we analyze the performance of the variants of our model with different directional SSM modules. As the results demonstrate, the bidirectional variants (forward and backward SSM modules together) overall show better performance (especially in the large dataset VGGSound) than the forward-only variant.

Impact of direction of conv1D. Here, we conduct a controlled experiment between two bidirectional methods: **AuM-Fo-Bi** and **AuM-Bi-Bi**, where the only difference is the presence of an additional backward Conv1D. As shown in Table 2, our design choice, which omits the backward Conv1D, generally yields better performance. We hypothesize that processing a single input sequence (the output from only the forward Conv1D) is more effective and natural for scanning in both forward and backward directions to understand entire context, compared to providing separate inputs to each directional SSM module and scanning in only one direction according to the input.

Impact of the class token position. Our extensive experiments reveal that positioning the class token in the middle of the sequence is the most suitable choice for our design.

Model	AudioSet (mAP)	AS-20K (mAP)	VGGSound (Acc.)	VoxCeleb (Acc.)	Speech Comm. V2 (Acc.)	Epic-Sounds (Acc.)
AST-S	**40.32** ± 0.08	**29.20** ± 0.11	**49.51** ± 0.06	39.70 ± 1.83	97.38 ± 0.07	52.42 ± 0.11
AuM-S (c)	39.68 ± 0.06 (-0.64)	28.89 ± 0.20 (-0.31)	49.43 ± 0.18 (-0.07)	**40.58** ± 1.11 (+0.89)	**97.51** ± 0.08 (+0.13)	**52.90** ± 0.40 (+0.48)

Table 3. **Results of Imagenet pretrained initializations of AST and AuM small models across various datasets.** Note that the setup of AuM-S is (c) in Fig. 1 due to the unavailability of the ViM weights for our preferred setup (b).

Model	VGGSound (Acc.)	VoxCeleb (Acc.)	Speech Comm. V2 (Acc.)	Epic-Sounds (Acc.)
AST-B/16	44.17 ± 0.14	**46.25** ± 1.08	90.37 ± 0.06	46.62 ± 0.04
AuM-B/16	**46.61** ± 0.18 (+2.44)	40.72 ± 1.11 (-5.53)	**94.78** ± 0.04 (+4.41)	**48.18** ± 0.13 (+1.57)

Table 4. **Results of Audioset pretrained initializations of AST and AuM base models across various datasets.**

However, it is important to note that the position of the class token is a crucial decision, as each variant exhibits a different optimal location for its use, which greatly impacts performance. An additional observation is that a forward-only SSM collapses when the class token is placed at the beginning of the sequence (head class token). This outcome is expected, as the information in the sequence following the class token is not incorporated into the class token.

3.5. Impact of Pre-Training

Out-of-domain pre-training. Initializing audio models with ImageNet pre-trained weights has become popular for audio classification [5, 10]. Specifically, AST demonstrates a significant performance improvement over training from scratch by utilizing the weights of a Supervised ImageNet pretrained ViT model. As presented in Table 1, our main results exclude models with pretraining (weight initialization) to provide a clear comparison between these two architectures. One might question why such results are not displayed. To the best of our knowledge, no released Vision Mamba Base model weights, comparable to ViT weights for the AST model, are available in the literature, preventing us from conducting this experiment directly. However, we aim to analyze both AuM and AST when initialized with out-of-domain pre-training weights. In this context, we utilize the only available Vision Mamba model, the small-sized ViM-S, to compare AuM-S and AST-S models. Despite the differences in architectural design with Vision Mamba, highlighted in Sections 2 and 3.4, where we note that the **AuM-Bi-Bi** variant is not the ideal choice for our AuM, the findings presented in Table 3 reveal that both models perform similarly. We believe that with the right weight initialization, our model could outperform AST, just as it does in scenarios without the use of vision domain pretrained weights.

From-scratch audio-only pre-training. After comparing AST and AuM by initializing them with weights from ImageNet pre-trained vision models, we also explore using AudioSet-trained weights of base models (from Table 1) as in-domain pre-training to initialize both AuM-B and AST-B. Here, unlike in the previous section, our model uses weights from a model that is architecturally identical to ours. The results, shown in Table 4, indicate that in-domain pre-training benefits both models, enhancing their performance. In this setting, AuM outperforms AST, except on the VoxCeleb dataset.

4. Conclusion

In this work, we introduce Audio Mamba (AuM), the first architecture for audio classification that is free from self-attention and purely based on state space models (SSM). Our extensive experiments highlight AuM's efficiency in terms of computational and memory use, as well as its competitive performance against the well-established Audio Spectrogram Transformers (AST). Considering its similarity to AST structure regarding patchifying the input spectrogram, adding positional embeddings, and processing the information sequentially but without costly self-attention, it shows great potential to become an alternative generic audio backbone. With the elimination of reliance on costly self-attention and the high efficiency of AuM in processing long sequence inputs, we believe that AuM brings an important contribution to the audio field for future potential applications. The ability to handle lengthy audio is increasingly crucial, especially with the rise of self-supervised multimodal learning [1, 2, 21] and generation that leverages in-the-wild data and Automatic Speech Recognition. Furthermore, AuM could be employed in self-supervised learning setups like Audio Masked Auto Encoders [3, 16] or multimodal learning tasks such as Audio-Visual pretraining [11, 12, 21] or Contrastive Language-Audio Pretraining [24, 29, 30].

References

[1] Hassan Akbari, Liangzhe Yuan, Rui Qian, Wei-Hong Chuang, Shih-Fu Chang, Yin Cui, and Boqing Gong. Vatt: Transformers for multimodal self-supervised learning from raw video, audio and text. In *NeurIPS*, 2021. 1, 5

[2] Jean-Baptiste Alayrac, Adria Recasens, Rosalia Schneider, Relja Arandjelović, Jason Ramapuram, Jeffrey De Fauw, Lucas Smaira, Sander Dieleman, and Andrew Zisserman. Self-

supervised multimodal versatile networks. In *NeurIPS*, 2020. 1, 5

[3] Alan Baade, Puyuan Peng, and David Harwath. MAE-AST: masked autoencoding audio spectrogram transformer. In *Proc. Interspeech*, 2022. 5

[4] Honglie Chen, Weidi Xie, Andrea Vedaldi, and Andrew Zisserman. VGGSound: A large-scale audio-visual dataset. In *Proc. ICASSP*, 2020. 2, 3

[5] Ke Chen, Xingjian Du, Bilei Zhu, Zejun Ma, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. HTS-AT: a hierarchical token-semantic audio transformer for sound classification and detection. In *Proc. ICASSP*, 2022. 5

[6] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, , Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling Egocentric Vision: Collection, Pipeline and Challenges for EPIC-KITCHENS-100. *IJCV*, 2022. 3

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proc. ICLR*, 2021. 1, 2

[8] Daniel Y Fu, Tri Dao, Khaled Kamal Saab, Armin W Thomas, Atri Rudra, and Christopher Re. Hungry hungry hippos: Towards language modeling with state space models. In *Proc. ICLR*, 2023. 1

[9] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. AudioSet: An ontology and human-labeled dataset for audio events. In *Proc. ICASSP*, 2017. 2, 3

[10] Yuan Gong, Yu-An Chung, and James Glass. AST: audio spectrogram transformer. In *Proc. Interspeech*, 2021. 1, 2, 3, 5

[11] Yuan Gong, Alexander H Liu, Andrew Rouditchenko, and James Glass. Uavm: Towards unifying audio and visual models. *IEEE Signal Processing Letters*, 2022. 1, 5

[12] Yuan Gong, Andrew Rouditchenko, Alexander H Liu, David Harwath, Leonid Karlinsky, Hilde Kuehne, and James Glass. Contrastive audio-visual masked autoencoder. In *Proc. ICLR*, 2022. 1, 5

[13] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023. 1, 2, 3

[14] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *Proc. ICLR*, 2022. 1

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 1

[16] Po-Yao Huang, Hu Xu, Juncheng Li, Alexei Baevski, Michael Auli, Wojciech Galuba, Florian Metze, and Christoph Feichtenhofer. Masked autoencoders that listen. In *NeurIPS*, 2022. 5

[17] Jaesung Huh, Jacob Chalk, Evangelos Kazakos, Dima Damen, and Andrew Zisserman. EPIC-SOUNDS: A Large-Scale Dataset of Actions that Sound. In *Proc. ICASSP*, 2023. 2, 3

[18] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley. Panns: Large-scale pretrained audio neural networks for audio pattern recognition. *TASLP*, 2020. 1

[19] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model. *arXiv preprint arXiv:2401.10166*, 2024. 1, 3

[20] Jun Ma, Feifei Li, and Bo Wang. U-mamba: Enhancing long-range dependency for biomedical image segmentation. *arXiv preprint arXiv:2401.04722*, 2024. 1

[21] Pedro Morgado, Yi Li, and Nuno Nvasconcelos. Learning representations from audio-visual spatial alignment. In *NeurIPS*, 2020. 1, 5

[22] Arsha Nagrani, Joon Son Chung, Weidi Xie, and Andrew Zisserman. VoxCeleb: Large-scale speaker verification in the wild. *Computer Speech & Language*, 2020. 2, 3

[23] Arsha Nagrani, Shan Yang, Anurag Arnab, Aren Jansen, Cordelia Schmid, and Chen Sun. Attention bottlenecks for multimodal fusion. In *NeurIPS*, 2021. 1

[24] Yi-Jen Shih, Hsuan-Fu Wang, Heng-Jui Chang, Layne Berry, Hung-yi Lee, and David Harwath. Speechclip: Integrating speech with pre-trained vision and language model. In *IEEE Spoken Language Technology Workshop (SLT)*, 2023. 1, 5

[25] Jimmy TH Smith, Andrew Warrington, and Scott W Linderman. Simplified state space layers for sequence modeling. In *Proc. ICLR*, 2023. 1

[26] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *Proc. ICML*, 2021. 1

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1

[28] Pete Warden. Speech Commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018. 2, 3

[29] Ho-Hsiang Wu, Prem Seetharaman, Kundan Kumar, and Juan Pablo Bello. Wav2clip: Learning robust audio representations from clip. In *Proc. ICASSP*, 2022. 1, 5

[30] Yusong Wu, Ke Chen, Tianyu Zhang, Yuchen Hui, Taylor Berg-Kirkpatrick, and Shlomo Dubnov. Large-scale contrastive language-audio pretraining with feature fusion and keyword-to-caption augmentation. In *Proc. ICASSP*, 2023. 1, 5

[31] Zhaohu Xing, Tian Ye, Yijun Yang, Guang Liu, and Lei Zhu. Segmamba: Long-range sequential modeling mamba for 3d medical image segmentation. *arXiv preprint arXiv:2401.13560*, 2024. 1

[32] Yijun Yang, Zhaohu Xing, and Lei Zhu. Vivim: a video vision mamba for medical video object segmentation. *arXiv preprint arXiv:2401.14168*, 2024. 3

[33] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. *arXiv preprint arXiv:2401.09417*, 2024. 1, 2, 3

Audio Mamba: Bidirectional State Space Model for Audio Representation Learning

Supplementary Material

Setting	Audioset	AS-20K	VGGSound	VoxCeleb	Speech Comm. V2	Epic Sounds
Optimizer	Adam(wd=5e-7,betas=(0.95, 0.999))					
Patch Size / Stride	16 x 16 / (16, 16)					
Batch Size	12					
Weighted Average	No					
Ensembling	No					
Loss Function	BCE			CE		
Multilabel	Yes		No			
Balanced Sampling	Yes	No				
Warm-up Duration	1000 steps					2 Epochs
Spectrogram Size	128x1024				128x128	128x1024
SpecAug (time / freq.)	48 / 192				48 / 48	48 / 192
Mixup	0.5		0		0.6	0.2
Epochs	5	25	20		30	
LR Sched. Type	MultiStepLR(start / step / decay)					LambdaLR*
LR Sched. Params	2 / 1 / 0.5	10 / 5 / 0.5	5 / 2 / 0.75		5 / 1 / 0.85	
Dataset Mean for Norm.	-4.268		5.077	-3.761	-6.846	No
Dataset Std. for Norm.	4.569		4.453	4.201	5.565	
Base LR	1e-5	5e-5	1e-5		2.5e-4	1e-5

Table 5. Training setup comparison across different datasets. Here, "*" indicates that we follow the official learning rate scheduler presented in the Epic Sounds paper.

5. Training Setup

We train all the models (AuM and AST) of all sizes (Base and Small) across six different datasets by following the training setup shown in Table 5.