

# Audio Mamba: Selective State Spaces for Self-Supervised Audio Representations

Sarthak Yadav<sup>1,2</sup>, Zheng-Hua Tan<sup>1,2</sup>

<sup>1</sup>Aalborg University, Denmark

<sup>2</sup>Pioneer Center for AI, Denmark

sarthaky@es.aau.dk, zt@es.aau.dk

## Abstract

Despite its widespread adoption as the prominent neural architecture, the Transformer has spurred several independent lines of work to address its limitations. One such approach is selective state space models, which have demonstrated promising results for language modelling. However, their feasibility for learning self-supervised, general-purpose audio representations is yet to be investigated. This work proposes Audio Mamba, a selective state space model for learning general-purpose audio representations from randomly masked spectrogram patches through self-supervision. Empirical results on ten diverse audio recognition downstream tasks show that the proposed models, pretrained on the AudioSet dataset, consistently outperform comparable self-supervised audio spectrogram transformer (SSAST) baselines by a considerable margin and demonstrate better performance in dataset size, sequence length and model size comparisons.

**Index Terms:** self-supervised learning, general-purpose audio representation learning, state space models

## 1. Introduction

In recent years, Transformers [1] and their successors have emerged as the go-to neural architecture for representation learning, traversing multiple domains and data modalities. This widespread adoption is a result of the generalization capabilities of the underlying scaled dot-product attention mechanism and transformers being inherently agnostic to input modality, as evidenced by several prominent works [2, 3]. The recent advancements in unsupervised and self-supervised learning have only further fueled this phenomenon, with the integration of masked predictive modeling and transformers spearheading several breakthroughs in natural language processing (NLP) [4], computer vision [5, 6, 7] and audio and speech processing [8, 9, 10, 11, 12]. However, the transformer architecture is not without its drawbacks. The quadratic complexity of the scaled dot-product attention operation at the heart of transformers scales poorly to very large sequences, and has spurred a lot of work on sub-quadratic approximations for attention [13, 14, 15] as well as token mixing approaches [16, 17].

More recently, a new class of approaches for addressing the shortcomings of transformers has emerged, called state space models (SSMs) [18, 19, 20, 21], which have demonstrated excellent long sequence modelling performance for a multitude of tasks and domains. Lying at the intersection of convolutional neural networks, recurrent neural networks and classical state space representations grounded in control theory, SSMs are sequence models governed by a set of first order differential equations. Several variants of SSMs have been proposed, ranging from (i) approaches that directly parameterize the continuous

space to model time-series data [22], (ii) methods that discretize SSM parameters and represent SSM computation as a convolution with a structured kernel, yielding state-of-the-art results on long sequence modelling tasks [18] and image and video recognition [19], and most recently, (iii) context-aware adaptations of SSMs enriched by a selection mechanism that perform well on language modelling, long context speech generation [21] as well as vision [23] tasks. Apart from improved generalization performance on several tasks and modalities and their causal nature, SSMs and their variants have several beneficial properties, including inherent resolution invariance [19], and exceptional long range modelling capabilities [18].

However, while SSMs and their variants have been investigated for a wide variety of domains and tasks, a thorough assessment of their ability to learn general-purpose audio representations without supervision is pending. In this work, we propose self-supervised Audio Mamba (SSAM), an approach at the intersection of SSMs and masked predictive modelling, for learning general-purpose audio representations from randomly masked spectrogram patches. SSAMs pretrained on AudioSet [24] outperform comparable self-supervised audio spectrogram transformer (SSAST) [10] based baselines on ten downstream audio recognition tasks, consistently yielding around 30% relative improvement in aggregate performance across several model configurations, despite having fewer number of parameters. Moreover, SSAMs adapt better to various patch sizes and input sequence lengths, while consistently performing better than SSASTs when pretrained with lower amounts of data. Code will be released publicly.

## 2. Method

### 2.1. Prerequisites: State space models

Structured state space sequence models (S4) [18] are a recent family of linear time invariant sequence models based on a continuous system that maps input  $x(t) \in \mathbb{R}$  to  $y(t) \in \mathbb{R}$  through a latent state  $h(t) \in \mathbb{R}^N$  using evolution parameter  $\mathbf{A}$  and projection parameters  $\mathbf{B}$  and  $\mathbf{C}$  as follows:

$$h'(t) = \mathbf{A}h(t) + \mathbf{B}x(t), \quad (1)$$

$$y(t) = \mathbf{C}h(t). \quad (2)$$

The above equations can be discretized through a discretization rule (usually a zero-order hold), with an additional timescale parameter  $\Delta$ :

$$\bar{\mathbf{A}} = \exp(\Delta\mathbf{A}), \quad (3)$$

$$\bar{\mathbf{B}} = (\Delta\mathbf{A})^{-1}(\exp(\Delta\mathbf{A}) - \mathbf{I}) \cdot \Delta\mathbf{B} \quad (4)$$

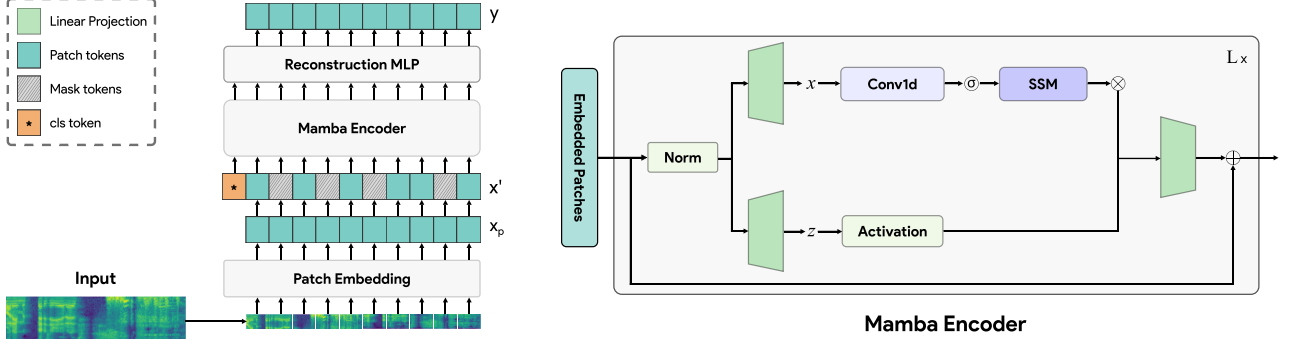


Figure 1: An overview of the proposed SSAM approach (left), and the constituent Mamba blocks (right).

Thus, S4 is effectively a discretized version of equations (1) and (2):

$$h_t = \bar{A}h_{t-1} + \bar{B}x_t, \quad (5)$$

$$y_t = Ch_t. \quad (6)$$

Finally, the model  $\text{SSM}(\bar{A}, \bar{B}, C)(\cdot)$  can be computed as a global convolution between input sequence  $\mathbf{x}$  and kernel  $\bar{\mathbf{K}} \in \mathbb{R}^M$ :

$$\bar{\mathbf{K}} = (\bar{C}\bar{B}, \bar{C}\bar{A}\bar{B}, \dots, \bar{C}\bar{A}^{M-1}\bar{B}), \quad (7)$$

$$\mathbf{y} = \mathbf{x} * \bar{\mathbf{K}}, \quad (8)$$

where  $M$  is the length of the input  $\mathbf{x}$ . It's worth noting that in the above equations (5-8), the parameters for the S4 model are not conditioned on the input and are time-invariant. This is in contrast to Mamba [21], where parameters  $\mathbf{B}, \mathbf{C} \in \mathbb{R}^{B \times L \times N}$  and  $\Delta \in \mathbb{R}^{B \times L \times D}$  are instead functions of the input  $\mathbf{x} \in \mathbb{R}^{B \times L \times D}$  and are context-aware, thus earning the moniker *selective* structured state spaces.

## 2.2. Self-Supervised Audio Mamba: SSAM

Figure 1 (left) shows an overview of the proposed Self-Supervised Audio Mamba (SSAM) approach.

**Creating patches and random masking:** For input spectrogram  $\mathbf{x} \in \mathbb{R}^{T \times F}$ , we compute non-overlapping patches of shape  $t \times f$ , yielding  $\mathbf{x}_p \in \mathbb{R}^{N \times (t \cdot f)}$  patches, where  $N$  is the number of patches. We then flatten these patches and project them linearly to a  $\mathbb{R}^{N \times d_m}$  dimensional space, followed by adding fixed sinusoidal positional embeddings for encoding positional information. We then add a representative class token to the beginning of the sequence, similar to [2, 10]. We then proceed to randomly mask 50% of the input patches using an unstructured masking strategy, which was demonstrated by [10] to work better for audio tasks as compared to frame-based patching, and replace these masked patches with a learnable *mask* token. Thus, input to the encoder is

$$\mathbf{x}' = [\text{cls}, \mathbf{x}_p^1, \mathbf{x}_p^2, \dots, \mathbf{x}_p^N] + E_{\text{pos}} \quad (9)$$

**Encoding:** We now feed these partially masked patches to the Mamba [21] encoder, as shown in Figure 1 (right). Mamba blocks internally expand the  $d_m$  dimensional input patches by an expansion factor  $E$ , projecting them back to  $d_m$  dimensions. The original Mamba paper [21] uses an expansion factor  $E = 2$ , which effectively means that every 2 Mamba blocks

correspond to the same number of parameters as those of a standard Transformer block. To facilitate comparison with standard ViT encoder configurations, we instead use a “wider” Mamba block: with an expansion factor  $E = 3$  and larger internal dimensions ( $d_{\text{state}} = 24, d_{\text{conv}} = 4$ ). This results in Mamba blocks that are closer to the number of parameters in a single transformer block and removes depth as a factor that might impact performance. This process yields encoded representations  $\mathbf{z} = \text{enc}(\mathbf{x}'), \mathbf{z} \in \mathbb{R}^{(N+1) \times d_m}$ .

**Reconstruction:** After encoding, a single hidden layer MLP is used to reconstruct patches from encoded representation  $\mathbf{z}$ :

$$\mathbf{y}' = \text{Linear}_{(t \cdot f)}(\sigma(\text{Linear}_{d_m}(\mathbf{z}))), \quad (10)$$

where  $\text{Linear}_d$  is a parameterized linear projection to dimensions  $d$ , and  $\sigma$  denotes the GELU non-linear activation function [25]. Removing the *cls* token from  $\mathbf{y}'$ , we get the reconstructed output  $\mathbf{y} \in \mathbb{R}^{N \times (t \cdot f)}$ . For pretraining, we use mean-squared error between the original input patches  $\mathbf{x}_p$  and the predicted reconstructions  $\mathbf{y}$ . This is in contrast to [10], which used a joint discriminative and reconstruction objective for pretraining. During downstream evaluation, random masking is removed and the reconstruction network is discarded, and the latent representation  $\mathbf{z}$  is used. Further details can be found in Section 3.2.

## 3. Experimental setup

### 3.1. Datasets

**Pretraining:** For pretraining, we use the AudioSet dataset [24] (AS) which has roughly 2 million 10-second weakly labelled YouTube clips that span an ontology of 527 classes. With over 5000 hours of audio data, AudioSet is widely used for audio representation learning [26, 10, 11].

**Downstream Evaluation:** Recently, the HEAR benchmark, which consists of 19 varied tasks from several audio domains, was proposed for thorough and systematic evaluation of audio representations. However, given the redundancy and demonstrated correlation in model performance amongst the proposed tasks [27], to avoid doing excessive evaluations we utilize a subset of HEAR that consists of the following ten diverse tasks: Beijing Opera, Crema-D, ESC-50, LibriCount, Mridangam Stroke and Tonic, NSynth Pitch 5h, Speech Commands 5h, FSD50K and VoxLingua107.

### 3.2. Implementation details

**Spectrogram features:** All datasets used are sampled at 16000 Hz. Log-scaled mel spectrograms with a window size of 25 ms,

Table 1: Comparing SSAMs with popular self-supervised audio representations on the evaluated downstream tasks. We used pre-trained models from cited papers to extract fixed feature vectors and conducted our own downstream experiments. For better 1v1 comparison, we trained directly comparable SSAST and SSAM models (highlighted with similar color levels). LS, AS, VP, LL stand for LibriSpeech, AudioSet, VoxPopuli and LibriLight datasets, respectively. \*MW-MAE parameter count includes decoder parameters, which is discarded after pretraining.

| Model                  | Data  | # Params | BO       | CD       | ESC-50   | LC       | Mri-S    | Mri-T    | NS-5h    | SC-5h    | F50K     | VL       | $s(m)$   |
|------------------------|-------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| <b>Naive Baselines</b> |       |          |          |          |          |          |          |          |          |          |          |          |          |
| HEAR-Naive [27]        | -     | -        | 52.6±2.4 | 30.9±0.8 | 5.8±0.2  | 33.5±1.1 | 38.0±1.3 | 36.4±1.9 | 18.6±4.4 | 8.5±0.4  | 7.1±0.2  | 11.2±0.5 | 5.2±0.8  |
| <b>Supervised</b>      |       |          |          |          |          |          |          |          |          |          |          |          |          |
| PaSST-Base [28]        | AS    | 86 M     | 94.9±0.5 | 61.0±0.3 | 94.8±0.3 | 60.1±0.2 | 96.5±0.1 | 87.6±0.6 | 23.3±0.9 | 66.6±1.4 | 64.2±0.1 | 25.5±0.8 | 74.7±0.4 |
| <b>SSL</b>             |       |          |          |          |          |          |          |          |          |          |          |          |          |
| W2V2-base [8]          | LS    | 94.4 M   | 74.0±1.0 | 46.4±0.3 | 31.1±0.4 | 51.2±0.2 | 77.3±0.2 | 55.1±0.3 | 7.4±0.8  | 90.8±0.3 | 18.1±0.1 | 35.5±0.8 | 44.1±0.2 |
| W2V2-large [8]         | VP    | 315.4 M  | 93.1±0.7 | 66.9±0.4 | 60.1±0.5 | 62.4±0.3 | 93.9±0.1 | 77.4±0.2 | 42.0±1.0 | 87.6±0.5 | 34.2±0.1 | 53.6±1.0 | 75.1±0.4 |
| WavLM-base [12]        | LS    | 94.4 M   | 89.4±0.7 | 56.3±0.2 | 46.6±0.4 | 63.2±0.3 | 95.1±0.1 | 83.4±0.2 | 37.3±0.8 | 57.2±0.8 | 29.9±0.1 | 22.6±0.6 | 61.7±0.2 |
| WavLM-large [12]       | Mix   | 315.4 M  | 96.4±0.5 | 57.2±0.2 | 47.9±0.4 | 61.1±0.3 | 96.8±0.1 | 89.5±0.1 | 53.7±0.5 | 46.2±0.8 | 29.0±0.1 | 23.7±0.9 | 65.1±0.2 |
| HuBERT-base [9]        | LS    | 94.4 M   | 92.1±0.6 | 70.8±0.2 | 57.8±0.6 | 56.5±0.3 | 94.4±0.1 | 84.9±0.3 | 19.4±0.7 | 93.2±0.1 | 32.3±0.1 | 61.8±0.6 | 74.5±0.2 |
| HuBERT-large [9]       | LL    | 315.4 M  | 94.1±0.7 | 70.7±0.1 | 60.3±0.4 | 59.9±0.2 | 95.3±0.1 | 83.5±0.3 | 19.3±0.8 | 83.2±0.7 | 31.5±0.1 | 66.1±0.9 | 74.4±0.3 |
| BEATs-iter3 [29]       | AS    | 90 M     | 94.0±0.8 | 67.3±0.2 | 83.7±0.3 | 68.0±0.2 | 94.7±0.1 | 95.8±0.1 | 69.4±0.8 | 85.2±0.3 | 53.6±0.2 | 38.5±1.0 | 87.6±0.3 |
| AudioMAE [11]          | AS    | 86 M     | 93.7±0.6 | 68.2±0.2 | 60.6±0.4 | 42.2±0.2 | 89.2±0.2 | 86.6±0.2 | 64.5±0.8 | 28.6±1.5 | 37.9±0.1 | 29.7±1.0 | 64.3±0.3 |
| MWMAE-Tiny [30]        | AS    | 12.6 M*  | 93.3±1.0 | 64.4±0.2 | 71.9±0.5 | 65.5±0.3 | 97.1±0.1 | 97.6±0.1 | 68.1±0.4 | 77.0±0.6 | 43.4±0.1 | 28.6±1.1 | 80.7±0.3 |
| MWMAE-Base [30]        | AS    | 92.5 M*  | 96.0±0.5 | 73.1±0.3 | 81.2±0.4 | 68.8±0.2 | 97.4±0.1 | 97.9±0.1 | 69.3±0.6 | 90.9±0.2 | 51.2±0.2 | 44.2±0.9 | 91.3±0.2 |
| <b>SSAST Based</b>     |       |          |          |          |          |          |          |          |          |          |          |          |          |
| SSAST [10]             | AS+LS | 89 M     | 93.4±0.9 | 56.5±0.2 | 68.4±0.4 | 60.7±0.3 | 96.7±0.1 | 96.3±0.1 | 66.8±0.7 | 53.5±1.3 | 38.2±0.1 | 28.5±0.9 | 73.1±0.2 |
| SSAST-Tiny             | AS    | 5.4 M    | 90.4±0.7 | 46.9±0.2 | 42.4±0.6 | 42.7±0.2 | 95.7±0.1 | 94.3±0.1 | 61.2±0.5 | 50.6±1.6 | 24.6±0.1 | 13.8±1.0 | 56.0±0.2 |
| SSAST-Small            | AS    | 21.5 M   | 93.2±0.5 | 51.6±0.2 | 50.1±0.6 | 50.0±0.3 | 96.2±0.1 | 95.0±0.1 | 63.8±0.4 | 58.3±1.2 | 31.6±0.1 | 15.6±0.7 | 63.4±0.3 |
| SSAST-Base             | AS    | 85.7 M   | 93.1±0.7 | 56.0±0.4 | 59.6±0.7 | 52.9±0.3 | 96.6±0.1 | 96.2±0.2 | 64.6±0.8 | 66.1±1.0 | 37.5±0.1 | 19.2±0.9 | 69.2±0.3 |
| <b>Proposed</b>        |       |          |          |          |          |          |          |          |          |          |          |          |          |
| SSAM-Tiny              | AS    | 4.8 M    | 93.7±0.8 | 61.8±0.3 | 70.6±0.2 | 59.2±0.4 | 97.1±0.1 | 94.9±0.1 | 62.0±0.7 | 74.8±0.4 | 41.3±0.2 | 27.8±1.0 | 76.3±0.2 |
| SSAM-Small             | AS    | 17.9 M   | 94.0±0.7 | 67.5±0.2 | 78.7±0.6 | 60.5±0.3 | 97.5±0.1 | 96.7±0.1 | 66.3±0.8 | 83.7±0.3 | 48.5±0.1 | 39.6±0.7 | 84.4±0.3 |
| SSAM-Base              | AS    | 69.3 M   | 93.2±1.1 | 70.3±0.2 | 81.0±0.3 | 63.5±0.2 | 97.7±0.1 | 96.9±0.1 | 70.5±0.5 | 87.9±0.3 | 52.2±0.1 | 50.4±0.7 | 89.7±0.3 |

a hop size of 10 ms and  $F = 80$  mel-spaced frequency bins, standardized on a per-instance basis, are used.

**Pretraining:** Our default configuration consists of  $l = 12$  number of stacked Mamba blocks with a model feature dimension  $d_m = 192$ , same as those of ViT-Tiny [2] encoder. We also evaluate Small ( $d_m=384$ ,  $l=12$ ) and Base ( $d_m=768$ ,  $l=12$ ) encoder configurations. All our models accept a [200, 80]-dim (time and frequency, resp.) input, corresponding to a randomly cropped 2-second audio clip during pretraining. By default, we use a patch embedding layer that computes  $(4 \times 16)$  shaped non-overlapping patches. All SSAM and their SSAST counterparts are trained for 100 epochs with a batch size of 1024 and a weight decay of 0.05. AdamW optimizer with a linear warmup for 10 epochs followed by a cosine learning rate decay schedule is used. No data augmentations were used.

**Downstream evaluation:** After pretraining, for every input audio clip we extract fixed sized feature vectors independent of the input audio duration inline with the HEAR protocol by extracting features on 2-second audio chunks, concatenating and taking their mean over time. We then train a single hidden layer MLP classifier with 1024 neurons for each task, using the official *hear-eval-kit* accompanying the HEAR benchmark. All experiments are repeated with 10 different random seeds.

**Aggregated Performance Metric:** When evaluating multiple audio representations on a varied list of tasks, several previous works, such as SUPERB [31], have proposed using an aggre-

gated performance metric that takes the difficulty levels of different tasks into account. Given the domain overlap, we use the aggregated normalized score as proposed by [30]. Specifically, for a model  $m$ , overall score  $s(m) \in [0, 100]$  is given as:

$$s(m) = \frac{1}{|T|} \sum_{t \in T} \frac{x_t(m) - \min_t}{\max_t - \min_t} * 100 \quad (11)$$

where  $x_t(m)$  denotes performance of the model  $m$  on task  $t$ , and  $\min_t$  and  $\max_t$  represent the worst and the best performance across all models on the task, thus taking into account the relative performance amongst all evaluated representations.

## 4. Results

### 4.1. Comparison with existing works

Table 1 depicts how the proposed SSAM models fare against recent audio representations. We pretrained the highlighted configurations from scratch, and they are colour coded to represent direct comparisons: they represent methods of similar complexity and identical feature embedding size. It's worth noting that the sizes of the feature vectors extracted from other referred methods can vary, which is inline with recent frameworks for evaluating self-supervised audio representations [27, 31]. While this might not be optimal, it is infeasible to retrain all pretrained representations to have the same embedding sizes,

which also undermines the utility of training such large models. In the table SSAST [10] represents the official SSAST released model, which was pretrained with the masked prediction + reconstruction multitask objective. We can see that all the proposed SSAM models improve over their SSAST counterparts by considerable margin, offering an absolute improvement of 20 points or over in the overall score. Moreover, both SSAM-Tiny and SSAM-Small outperform the official SSAST [10] baseline in the overall score, despite having fewer parameters and only being trained with the reconstruction objective. SSAM-Base model, with an overall score of  $89.7 \pm 0.3$ , considerably outperforms the the SSAST-Base configuration as well as the official SSAST [10] model while having fewer parameters. The proposed SSAMs outperform all evaluated methods apart from Multi-Window Masked Autoencoders (MWMAEs) [30], which are the current top performers for the selected evaluation protocol. However, it's worth noting that unlike SSAMs, MWMAEs are not causal. Investigating Mamba within a Masked Autoencoder (MAE) [7] framework is far from trivial due to architectural nuances and would be the focus of future work. Overall, SSAMs outperform comparable standard transformer based approaches by a considerable margin and performs very favourably compared to popular audio representations, highlighting the effectiveness of selective state space models for learning general-purpose audio representations.

#### 4.2. Ablations

**Patch Size:** We investigate how SSAM fare against standard SSASTs with changing number of input patches. To this end, we pretrain models (Tiny configuration only) with the following 3 patch size sizes: (4, 8), (4, 16), and (8, 16). These settings

Table 2: Patch size ablations with the ViT-Tiny configuration

| Model      | Patch Size | # Patches | $s(m)$                           |
|------------|------------|-----------|----------------------------------|
| SSAST-Tiny | (8, 16)    | 125       | $47.3 \pm 0.3$                   |
| SSAM-Tiny  | (8, 16)    | 125       | <b><math>58.2 \pm 0.3</math></b> |
| SSAST-Tiny | (4, 16)    | 250       | $56.0 \pm 0.2$                   |
| SSAM-Tiny  | (4, 16)    | 250       | <b><math>76.3 \pm 0.2</math></b> |
| SSAST-Tiny | (4, 8)     | 500       | $54.1 \pm 0.4$                   |
| SSAM-Tiny  | (4, 8)     | 500       | <b><math>76.0 \pm 0.2</math></b> |

not only represent differing sequence lengths, but also different frequency and time resolution scenarios. From Table 2, we can observe that the proposed SSAM approach consistently outperforms SSAST baselines for all evaluated patch sizes. Moreover, the performance gap between the two approaches grows as the number of patches increases, with the absolute difference in  $s(m)$  between SSAM and SSAST increasing from 10.9 to 20.3 between the (8, 16) and (4, 16) settings. Overall, these experiments highlight the better ability of selective state space models to adapt to different frequency and time resolutions.

**Amount of pretraining data used:** To simulate how both methodologies fare with the amount of data used for pretraining, we conduct ablations where we pretrain the default model with different fractions of the AudioSet corpus (Table 3). It is evident from Table 3 that SSAMs scale much better with the amount of pretraining data, as the absolute difference in  $s(m)$  between SSAMs and SSASTs continues to increase as more and more data is used for pretraining, highlighting the data efficacy of the proposed approach.

Table 3: Overall downstream performance v/s amount of pre-training data used

| Model      | Pretraining data | $s(m)$                           |
|------------|------------------|----------------------------------|
| SSAST-Tiny | 10% of AS        | $53.6 \pm 0.4$                   |
| SSAM-Tiny  | 10% of AS        | <b><math>62.5 \pm 0.2</math></b> |
| SSAST-Tiny | 25% of AS        | $52.9 \pm 0.3$                   |
| SSAM-Tiny  | 25% of AS        | <b><math>70.5 \pm 0.3</math></b> |
| SSAST-Tiny | 100% of AS       | $56.0 \pm 0.2$                   |
| SSAM-Tiny  | 100% of AS       | <b><math>76.3 \pm 0.2</math></b> |

**Bidirectional SSM:** Given the causal, unidirectional nature of SSAM, which is unlike the attention mechanism in a transformer that learns relevance amongst all its input tokens, a natural question arises: would bidirectional computation of SSMs yield better results? [23] attempted to address this question

Table 4: Is bidirectional modelling helpful?

| Model      | SSM Type | $s(m)$                           |
|------------|----------|----------------------------------|
| SSAST-Tiny | None     | $56.0 \pm 0.2$                   |
| SSAM-Tiny  | Mamba    | <b><math>76.3 \pm 0.2</math></b> |
| SSAM-Tiny  | Vim      | $65.7 \pm 0.3$                   |

by proposing the *Vim* block that ingests image patches with two separate Mamba subbranches operating in the forward and the backward direction, which improved performance considerably for semantic segmentation. Table 4 shows ablations with bidirectional computation by replacing our underlying Mamba blocks with Vim [23] blocks in our default configuration. We can see that while the bidirectional Vim block performs better than the standard Transformer block, it performs much worse than a unidirectional Mamba block by a considerable margin. This makes intuitive sense, given that acoustic elements of interest in a spectrogram representation move unidirectionally in time, unlike images, where objects of interest lie in a coordinate plane. We thus conclude that, as opposed to image patches, bidirectional modelling with selective state spaces is not needed for audio recognition from spectrogram patches.

## 5. Conclusion

This work presents Self-Supervised Audio Mamba (SSAM), a selective structured state space model based on Mamba for learning general-purpose audio representations from masked audio spectrograms through self-supervision. Using the AudioSet corpus for pretraining, we evaluate SSAMs against directly comparable Self-supervised Audio Spectrogram Transformer (SSAST) baselines on ten varied downstream audio recognition tasks in a multitude of settings. Following thorough empirical analysis, we conclude that SSAM models outperform baseline SSASTs by a large margin, attaining an absolute improvement of over 20 points in aggregate performance. SSAMs consistently perform better than their standard transformer-based counterparts when using less amount of pretraining data, adapt better to changing number of input sequence lengths, and scale better with models of different sizes, establishing the viability of state space models for learning general-purpose audio representations.

## 6. 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," *Advances in neural information processing systems*, vol. 30, 2017.
- [2] 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 *International Conference on Learning Representations*, 2021.
- [3] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli, "data2vec: A general framework for self-supervised learning in speech, vision and language," in *Proceedings of the 39th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, Eds., vol. 162. PMLR, 17–23 Jul 2022, pp. 1298–1312.
- [4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.
- [5] Z. Xie, Z. Zhang, Y. Cao, Y. Lin, J. Bao, Z. Yao, Q. Dai, and H. Hu, "Simmim: A simple framework for masked image modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9653–9663.
- [6] H. Bao, L. Dong, S. Piao, and F. Wei, "BEit: BERT pre-training of image transformers," in *International Conference on Learning Representations*, 2022.
- [7] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 000–16 009.
- [8] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 12 449–12 460.
- [9] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhota, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE Transactions on Audio, Speech, and Language Processing*, pp. 1–1, 2021.
- [10] Y. Gong, C.-I. Lai, Y.-A. Chung, and J. Glass, "Ssast: Self-supervised audio spectrogram transformer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 10 699–10 709.
- [11] P.-Y. Huang, H. Xu, J. Li, A. Baevski, M. Auli, W. Galuba, F. Metze, and C. Feichtenhofer, "Masked autoencoders that listen," *Advances in Neural Information Processing Systems*, vol. 35, pp. 28 708–28 720, 2022.
- [12] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.
- [13] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, "Transformers are rns: Fast autoregressive transformers with linear attention," in *International conference on machine learning*. PMLR, 2020, pp. 5156–5165.
- [14] S. Wang, B. Z. Li, M. Khabsa, H. Fang, and H. Ma, "Linformer: Self-attention with linear complexity," *arXiv preprint arXiv:2006.04768*, 2020.
- [15] K. M. Choromanski, V. Likhoshesterov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Q. Davis, A. Mohiuddin, L. Kaiser, D. B. Belanger, L. J. Colwell, and A. Weller, "Rethinking attention with performers," in *International Conference on Learning Representations*, 2021.
- [16] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in neural information processing systems*, vol. 34, pp. 24 261–24 272, 2021.
- [17] F. Mai, J. Zuluaga-Gomez, T. Parcollet, and P. Motlicek, "Hyper-Conformer: Multi-head HyperMixer for Efficient Speech Recognition," in *Proc. INTERSPEECH 2023*, 2023, pp. 2213–2217.
- [18] A. Gu, K. Goel, and C. Re, "Efficiently modeling long sequences with structured state spaces," in *International Conference on Learning Representations*, 2022.
- [19] E. Nguyen, K. Goel, A. Gu, G. Downs, P. Shah, T. Dao, S. Baccus, and C. Ré, "S4ND: Modeling images and videos as multidimensional signals with state spaces," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022.
- [20] D. Y. Fu, T. Dao, K. K. Saab, A. W. Thomas, A. Rudra, and C. Re, "Hungry hungry hippos: Towards language modeling with state space models," in *The Eleventh International Conference on Learning Representations*, 2023.
- [21] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.
- [22] M. Zhang, K. K. Saab, M. Poli, T. Dao, K. Goel, and C. Re, "Effectively modeling time series with simple discrete state spaces," in *The Eleventh International Conference on Learning Representations*, 2023.
- [23] L. Zhu, B. Liao, Q. Zhang, X. Wang, W. Liu, and X. Wang, "Vision mamba: Efficient visual representation learning with bidirectional state space model," *arXiv preprint arXiv:2401.09417*, 2024.
- [24] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 776–780.
- [25] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.
- [26] A. Saeed, D. Grangier, and N. Zeghidour, "Contrastive learning of general-purpose audio representations," in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 3875–3879.
- [27] J. Turian, J. Shier, H. R. Khan, B. Raj, B. W. Schuller, C. J. Steinmetz, C. Malloy, G. Tzanetakis, G. Velarde, K. McNally, M. Henry, N. Pinto, C. Noufi, C. Clough, D. Herremans, E. Fonseca, J. Engel, J. Salamon, P. Esling, P. Manocha, S. Watanabe, Z. Jin, and Y. Bisk, "HEAR: Holistic Evaluation of Audio Representations," in *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*. PMLR, Jul. 2022, pp. 125–145, ISSN: 2640-3498.
- [28] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient Training of Audio Transformers with Patchout," in *Proc. Interspeech 2022*, 2022, pp. 2753–2757.
- [29] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, W. Che, X. Yu, and F. Wei, "BEATs: Audio pre-training with acoustic tokenizers," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 5178–5193.
- [30] S. Yadav, S. Theodoridis, L. K. Hansen, and Z.-H. Tan, "Masked autoencoders with multi-window local-global attention are better audio learners," in *The Twelfth International Conference on Learning Representations*, 2024.
- [31] S.-w. Yang, P.-H. Chi, Y.-S. Chuang, C.-I. Jeff Lai, K. Lakhota, Y. Y. Lin, A. T. Liu, J. Shi, X. Chang, G.-T. Lin, T.-H. Huang, W.-C. Tseng, K.-t. Lee, D.-R. Liu, Z. Huang, S. Dong, S.-W. Li, S. Watanabe, A. Mohamed, and H.-y. Lee, "SUPERB: Speech Processing Universal PERFORMANCE Benchmark," in *Proc. Interspeech 2021*, 2021, pp. 1194–1198.