MASKED SPECTROGRAM PREDICTION FOR SELF-SUPERVISED AUDIO PRE-TRAINING

Dading Chong\textsuperscript{1,\dagger}, Helin Wang\textsuperscript{2,\dagger,∗}, Peilin Zhou\textsuperscript{3}, Qingcheng Zeng\textsuperscript{4}

\textsuperscript{1}Peking University, Shenzhen, China
\textsuperscript{2}Johns Hopkins University, Baltimore, USA
\textsuperscript{3}Hong Kong University of Science and Technology, Guangzhou, China
\textsuperscript{4}Department of Linguistics, Northwestern University, USA

ABSTRACT

Transformer-based models attain excellent results and generalize well when trained on sufficient amounts of data. However, constrained by the limited data available in the audio domain, most transformer-based models for audio tasks are finetuned from pre-trained models in other domains (e.g., image), which has a notable gap with the audio domain. Other methods explore the self-supervised learning approaches directly in the audio domain but currently do not perform well in the downstream tasks. In this paper, we present a novel self-supervised learning method for transformer-based audio models, called masked spectrogram prediction (MaskSpec), to learn powerful audio representations from unlabeled audio data (AudioSet used in this paper). Our method masks random patches of the input spectrogram and reconstructs the masked regions with an encoder-decoder architecture. Experimental results demonstrate MaskSpec reaches the performance of 0.471 (mAP) on AudioSet, 0.854 (mAP) on OpenMIC2018, 0.982 (accuracy) on ESC-50, 0.976 (accuracy) on SCV2, and 0.823 (accuracy) on DCASE2019 Task1A. The source code and pre-trained models have been released.\textsuperscript{1}

Index Terms— Self-supervised learning, transformer, masked auto-encoder, pre-trained model.

1 INTRODUCTION

The recent research demonstrates that the inductive biases in convolution operation can enable strongly sample-efficient training with limited data [1]. However, in the case of sufficient data available, inductive biases can be overly restrictive and restrict the upper limit of the model performance [2]. Instead, transformer-based models [3] which are based purely on attention without inductive biases have a higher ceiling, especially when extending to the downstream small-scaled datasets. On the premise of sufficient data and the same scale of parameters, transformer-based models achieve better results than convolutional neural networks (CNNs) [4] and recurrent neural networks (RNNs) [5] in various fields such as computer vision (CV) [2, 6], natural language processing (NLP) [7, 8] and automatic speech recognition (ASR) [9, 10]. Recently, transformer-based models are also considered in the detection and classification of acoustic scenes and events (DCASE) challenges [11, 12]. However, limited data available in this area becomes a bottleneck that restricts the development of pre-training a transformer-based model.

Currently, there are two existing strategies to alleviate this problem: (1) adapting weights from the pre-trained models of other domains (e.g., image) and (2) designing self-supervised learning methods to directly pre-train models with unlabeled audio data. For the first strategy, Gong et al. [11] initialized an audio spectrogram transformer (AST) with the weights of the data-efficient image transformer (Deit) [13] pre-trained on Imagenet [14], and performed incremental pre-training using AudioSet [15]. Koutini et al. [12] took the same approach, and finetuned the weights from Deit and vision transformer (ViT) [2] using AudioSet with various data augmentation methods. Both got the outstanding performance and outperformed the previous CNN-based methods [16, 17, 18, 19], but the effectiveness and transferability of transferring knowledge cross domains is still unclear due to the fundamental discrepancy between different domains. For instance, the channel numbers and the resolution of the inputs are hard to match between RGB images and Mel spectrograms. The second strategy adopts self-supervised learning with unlabeled audio data for pre-training. In [20], Baevski et al. explored to learn powerful speech representations from Librispeech and the larger LibriVox (LV-60k) [21]. In addition, Gong et al. [22] proposed to pre-train the AST model with joint discriminative and generative masked spectrogram patch modeling (MSPM) using unlabeled audio from AudioSet and Librispeech. While self-supervised learning can effectively reduce the dependence on the amount of data, the performance of self-supervised methods could not be equal to the performance of that adapts weights from other domain pre-trained models.

To overcome the above problems, in this paper, we investigate how to improve the performance of self-supervised learning...
pre-training with unlabeled audio data. Inspired by the success of mask autoencoder (MAE) proposed by He et al. [23] for image self-supervised learning, we present masked spectrogram prediction (MaskSpec), a pre-training objective that directly recovers the masked patches of spectrogram. More specifically, a certain percentage of patches within the input spectrogram are randomly masked and removed from the input of the encoder, and the objective is to refactor the information and position of the masked patches based on the remaining patches. In this way, the pre-trained model gains the ability to have an adequate understanding of the complex time-frequency structures within the spectrogram. To facilitate this research, we pre-train the audio spectrogram transformer model with MaskSpec on the largest open-source audio dataset (i.e. AudioSet [15]), and evaluate the model on five downstream tasks: audio tagging, environment sound classification, acoustic scene classification, polyphonic music instrument recognition and speech command recognition. Experimental results indicate that our proposed method outperforms both from-scratch self-supervised methods and cross-domain transferring methods.

2. MASKED SPECTROGRAM PREDICTION

As shown in Figure 1, our proposed self-supervised learning approach (MaskSpec) aims to reconstruct the masked patches of the spectrogram with an asymmetrical encoder-decoder architecture. We choose to use the spectrogram as the input to the model instead of using the raw waveform [20] or others for three reasons: (1) spectrogram is sparse and contains abundant low-level acoustic information, and it has similar characteristics as the image, which has been proven to successfully adapt the transformer-based models [2]. (2) spectrogram input provides the state-of-the-art results for many audio tasks [16, 24] (3) spectrogram can be directly used as the input, but raw waveform often needs extra convolutional layers, which causes more computational costs.

2.1. Masking strategy

Inspired by pre-training through Masked Language Modeling (MLM) [7] in natural language processing, the same random mask strategy is adopted in this paper. Though several other masking strategies (e.g. structured patchout [12]) have been proposed, we find the simple random mask strategy is effective and easy to implement. Given the spectrogram $T \in \mathcal{R}^{N_s \times N_f}$ (where $N_s$ and $N_f$ denote the number of frames and the frequency bin within one frame) extracted from a training sample of dataset $D_T$, a size of $p \times p$ sliding window with the same hop size is first applied to get the patches $E = \{e_1, ..., e_n\}$. Here, $n$ denotes the number of patches and $n = \left\lfloor \frac{N_s}{p} \right\rfloor \times \left\lfloor \frac{N_f}{p} \right\rfloor$. Let $N = \lceil n \times \alpha \rceil$ be the number of the masked patches, $N$ is determined by $n$ and a preset masking ratio $\alpha$, where $\alpha \in [0.05, 0.95]$ in our experiments. Note that different from the previous methods such as the masked patch sampling [22], we directly remove the masked patches to make the pre-training efficient and keeps the position index of all the patches for the decoder to do the reconstruction.

2.2. Encoder

To make a fair comparison, we adopt the same encoder architecture as PaSST [12], and another two scales of the encoder (i.e. PaSST-Small and PaSST-Tiny) have also been explored, which are called MaskSpec, MaskSpec-Small and MaskSpec-Tiny respectively. To be more specific, the MaskSpec model is composed of a learnable linear projection and a stack of $N_d = 12$ transformer blocks. In each transformer block, there are $N_h$ attention heads, $N_{emb}$ dimension of embedding and position-wise feed-forward network (FFN) with a hid-

---

**Algorithm 1** Self-supervised masked spectrogram prediction

**Require:** Unlabeled Audio dataset $D_T$, transformer-based encoder-decoder model $M$

**Input:** Set of patches $E$; Number of the whole patches $n$; Mask ratio $\alpha$

**Output:** Unmasked Patches $\hat{E}$; Masked Patches $\tilde{E}$; Set of position index $I$

**Init:** $i = 0$; $I = \{\}$; $\tilde{E} = \{\}$; $\hat{E} = \{\}$

1: $N = \lceil n \times \alpha \rceil$
2: for $i < N$ do
3: Get index $i \sim \text{unif}\{1, n\}$ and $i \notin I$
4: $I = I \cup i$; $\tilde{E} = \tilde{E} \cup e_i$
5: $\hat{E} = \mathbb{L}_E \tilde{E}$
6: return $\hat{E}$, $\tilde{E}$ and $I$

**MaskSpec ($D_T, M$)**

**Input:** $D_T, M; E; \alpha; n$

7: for every epoch do
8: $L_{epoch} = 0$
9: for $T \in D_T$ do
10: Split $T$ into $n$ patches $E = \{e_1, e_2, ..., e_N\}$
11: Add position embedding to $E$
12: $\tilde{E}, \hat{E}, I = \text{Random masking}(E, \alpha)$
13: $\tilde{O} = M_{encoder}(\tilde{E})$
14: Insert learnable vector $S$ to $\tilde{O}$ and get $O$
15: $Y = M_{decoder}(O)$
16: $L_{epoch} += L(\hat{E}, Y; \theta)$
17: minimize $L_{epoch}$ to update $M$

return $M$
Fig. 2. Results of finetuning pre-trained models with different mask ratios on AudioSet.

The decoder is only used during pre-training to perform the spectrogram reconstruction. Therefore, a relatively lightweight decoder [23] can be applied for efficiency. Specifically, the decoder contains 8 layers of transformer blocks and a linear projection layer. Each transformer block contains 16 attention heads with the embedding size of 512, and the feed-forward layers have a dimensionality of 2048. The function of the last layer is to convert the output of the final FFN to the masked patches, in which each patch has a dimensionality of \( p \times p \). According to the position index of the masked patches saved in the masking strategy, we insert shared and learnable vectors [2] into masking regions of the output of the encoder, and reassemble them into the same number of patches as the whole patches before masking. Then we inject information about the absolute position of the tokens in the sequence and feed them to the decoder. In this paper, the same decoder is used to reconstruct the masked patches for MaskSpec, MaskSpec-Small and MaskSpec-Tiny.

2.3. Decoder

The decoder is only used during pre-training to perform the spectrogram reconstruction. Therefore, a relatively lightweight decoder [23] can be applied for efficiency. Specifically, the decoder contains 8 layers of transformer blocks and a linear projection layer. Each transformer block contains 16 attention heads with the embedding size of 512, and the feed-forward layers have a dimensionality of 2048. The function of the last layer is to convert the output of the final FFN to the masked patches, in which each patch has a dimensionality of \( p \times p \). According to the position index of the masked patches saved in the masking strategy, we insert shared and learnable vectors [2] into masking regions of the output of the encoder, and reassemble them into the same number of patches as the whole patches before masking. Then we inject information about the absolute position of the tokens in the sequence and feed them to the decoder. In this paper, the same decoder is used to reconstruct the masked patches for MaskSpec, MaskSpec-Small and MaskSpec-Tiny.

2.4. Implementation of framework

The pseudo-code of the whole MaskSpec can be seen in Algorithm 1. As presented in Section 2.1, the input spectrogram \( T \) is split into \( n \) spectrogram patches, and we then add position information to them via sinusoidal position encoding. We randomly mask \( \alpha \) spectrogram patches, and the index of masked positions are denoted as \( I = \{ I_1, ..., I_N \} \). The rest of the patches \( \widehat{E} = \{ e_{i} \}_{i \notin I}^{n-N} \) are fed into the transformer encoder as described in Section 2.2. The output of the final hidden layers \( \widehat{O} = \{ o_{i} \}_{i \notin I}^{n-N} \) are the encoder representations of the input surviving patches. Next, we fill each masked patch with a learnable vector \( S \in \mathbb{R}^{N_{emb}} \), and get the input of the decoder \( O = \{ o_{1}, ..., o_{N} \} \). The transformer decoder and a final linear projection layer map \( O \) to the same dimension as the original masked patches \( \hat{E} \). The optimization target is to make the reconstructed patches \( \hat{Y} = \{ y_{1}, ..., y_{T} \} \) and masked patches \( \widehat{E} = \{ e_{1}, ..., e_{I_N} \} \) as close as possible in the Euclidean space. Thus, the mean squared error (MSE) loss function between the reconstructed patches and original

\[
\mathcal{L}(\hat{E}, Y; \theta) = \sum_{i=1}^{I_N} \| \hat{E}_i - Y_i \|^2
\]  

(1)

where \( \theta \) denotes the learnable parameters of the model.

3. EXPERIMENTS

3.1. Pre-training

In the self-supervised pre-training stage, we conducted experiments on widely-used large-scale audio dataset (AudioSet [15]) Following [12], a size of 128 \( \times \) 992 spectrogram is extracted. We take \( p = 16 \) as the patch size, so the spectrogram is split to \( 8 \times 62 \) patches. We randomly mask the patches at a certain percentage \( \alpha \) from 5% to 95%. Unless specifically stated, \( \alpha \) is set as 75%. Limited by the computational resource, the MaskSpec runs for 80 epochs, which takes about four days using 8 Nvidia Tesla V100 32GB GPU cards. The AdamW [27] optimizer with an initial learning rate of 0.001 and a weight decay of 0.05 is applied. The cosine decay learning rate scheduler [28] is used to warm up the training during the first 40 epochs.

3.2. Finetuning

After the self-supervised pre-training, we apply a linear layer with the dimension varies according to downstream tasks, and finetune all the parameters with the downstream datasets. We take audio tagging, environment sound classification, acoustic scene classification, polyphonic music instrument recognition, and speech command recognition as the downstream tasks to verify the effectiveness of the MaskSpec. For all downstream tasks, we use mixup [29] at both the waveform

| Model                  | Pre-train Settings       | mAP   | Param |
|------------------------|--------------------------|-------|-------|
| CNN10[16]              | random initialization    | 0.380 | 5.2M  |
| CNN14[16]              | random initialization    | 0.431 | 81M   |
| PSLA[25]               | random initialization    | 0.443 | 13.6M |
| PaSST-Tiny*            | random initialization    | 0.354 | 5.6M  |
| PaSST-Small*           | random initialization    | 0.413 | 22M   |
| PaSST*                 | random initialization    | 0.421 | 86M   |
| AST[11]                | IN-21K                   | 0.457 | 86M   |
| PaSST-Tiny*            | IN-21K                   | 0.397 | 5.6M  |
| PaSST-Small*           | IN-21K                   | 0.431 | 22M   |
| PaSST[12]              | IN-21K                   | 0.471 | 86M   |
| CNN14[26]              | AudioSet                 | 0.376 | 81M   |
| MaskSpec-Tiny          | AudioSet                 | 0.403 | 5.6M  |
| MaskSpec-Small         | AudioSet                 | 0.442 | 22M   |
| MaskSpec               | AudioSet                 | 0.471 | 86M   |

Table 1. Comparisons with previous works on AudioSet.

Here, IN-21K means ImageNet-21K [14]. * means this model was not implemented in the previous paper, and we implemented the model and keep the consistent training setups.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Table 2. Results on downstream tasks and comparisons with previous works. ⋄ means that the model is fine-tuned with AudioSet before applied to downstream tasks.

| Model                                      | Pre-train Settings | AudioSet-20K | ESC50 | DCASE2019 | OpenMIC18 | SCV2  |
|--------------------------------------------|--------------------|--------------|-------|-----------|-----------|-------|
| CNN14[16] ⋄                               | -                  | 0.278        | 0.947 | 0.764     | -         | -     |
| PSLA[25] ⋄                                | -                  | 0.319        | 0.877 | -         | -         | -     |
| AST[11] ⋄                                 | IN-21K & AudioSet  | 0.347        | 0.956 | -         | -         | 0.981 |
| PaSSST-Tiny* ⋄                            | IN-21K & AudioSet  | -            | 0.943 | 0.763     | 0.785     | 0.953 |
| PaSSST-Small* ⋄                           | IN-21K & AudioSet  | -            | 0.963 | 0.788     | 0.809     | 0.971 |
| PaSSST [12] ⋄                             | IN-21K & AudioSet  | -            | 0.968 | -         | 0.843     | -     |
| SSAST-Tiny [22]                            | Librispeech & AudioSet | 0.271     | 0.795 | -         | -         | 0.972 |
| SSAST-Small [22]                           | Librispeech & AudioSet | 0.308     | 0.854 | -         | -         | 0.977 |
| SSAST [22]                                 | Librispeech & AudioSet | 0.310     | 0.888 | -         | -         | 0.980 |
| MaskSpec-Tiny                              | AudioSet           | -            | 0.822 | 0.742     | 0.791     | 0.967 |
| MaskSpec-Small                             | AudioSet           | 0.289        | 0.907 | 0.804     | 0.818     | 0.973 |
| MaskSpec                                   | AudioSet           | 0.323        | 0.896 | 0.801     | 0.814     | 0.977 |
| MaskSpec-Tiny ⋄                            | AudioSet           | -            | 0.975 | 0.797     | 0.834     | 0.961 |
| MaskSpec-Small ⋄                           | AudioSet           | -            | 0.980 | 0.824     | 0.853     | 0.973 |
| MaskSpec ⋄                                 | AudioSet           | -            | 0.982 | 0.823     | 0.853     | 0.976 |

and spectrogram, and waveforms randomly rolling over time as the data augmentation [12]. Optimizers and learning strategies are the same as self-supervised stage, except for warming up for the first 5 epochs. Besides, layer-wise learning rate decay [30] is adopted following [7]. And we finetune the model for 80 epochs on AudioSet and 100 epochs for the other datasets. (1) **Audio Tagging:** We conducted experiments on Audioset [15]. Referring to [12], weight sampling is adopted to mitigate the negative effects of unbalanced distribution in our experiments. We have two settings for fine-tuning: (1) the full training data and (2) only the balanced data (AudioSet-20K). The widely-used metric mean average precision (mAP) is adopted for performance evaluation and comparison. (2) **Environment Sound Classification:** We use ESC-50[31] with the official 5-fold cross validation, and take the average accuracy of the 5 folders as the metric. (3) **Acoustic Scene Classification:** The DCASE2019 task1A dataset [32] contains 2-channel audio clips. Our model is finetuned with the left-channel, right-channel, and the average of them respectively. The accuracy of the ensemble results of the three is used as the evaluation metric. (4) **Polyphonic Musical Instrument Recognition:** We use the OpenMIC2018 dataset [33] and test its mAP values. (5) **Speech Command Recognition:** Speech Command V2 (SCV2) [34] is used and the accuracy is calculated.

### 3.3. Results and discussions

Table 1 reports the mAP of finetuning the encoder with full training data in AudioSet, and the result shows that the overall performance of transformer-based models is better than CNN-based models. We compare the same network structures (e.g. PaSSST has the same structure as MaskSpec) trained by random initialization, previous pre-training methods and our proposed method. Experiment results indicate that pre-training leads to great improvement of mAP against random initialization. Our proposed MaskSpec performs much better than another self-pretraining method [26], and even achieves comparable results to the cross-modal transfer methods [11, 12]. Figure 2 shows the influence of the masking ratio $\alpha$ by fine-tuning the encoder on AudioSet. We can see that effective self-supervised pre-training can be carried out within the range of $[15\%, 85\%]$, and the best result can be achieved with a ratio of 75%.

In Table 2, we comprehensively compare the performance of MaskSpec in various downstream tasks with other self-supervised and supervised methods. Compared with another self-supervised method (SSAST [22]), our proposed method has the stronger generalization in all downstream tasks, except that performs slightly worse than SSAST [22] on SCV2. This is because SSAST using extra Librispeech for pre-train, which is totally a speech-based dataset. The proposed method preforms worse than AST [11] on AudioSet-20K, which uses extra image data for pre-train. Besides, by finetuning on AudioSet before applied to downstream tasks, better performance can be obtained under all downstream tasks. Comparing with other supervised methods [16, 25, 11, 12], we find that MaskSpec can beat them in the downstream tasks without using extra data, indicating that the proposed MaskSpec brings better robustness and generalization. Among the results achieved by different-scaled models, we found an interesting phenomenon that PaSSST-Small achieved excellent results in all the tasks, sometimes even better than PaSSST. Thanks to such a self-supervised learning method, the relative small model can also perform well.
4. REFERENCES

[1] Stéphane d’Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun, “Convit: Improving vision transformers with soft convolutional inductive biases,” in International Conference on Machine Learning, PMLR, 2021, pp. 2286–2296.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al., “An image is worth 16x16 words: Transformers for image recognition at scale,” arXiv preprint arXiv:2010.11929, 2020.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, “Attention is all you need,” Advances in Neural Information Processing Systems, vol. 30, 2017.

[4] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi, “Understanding of a convolutional neural network,” in International Conference on Engineering and Technology (ICET). IEEE, 2017, pp. 1–6.

[5] Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Černocký, and Sanjeev Khudanpur, “Recurrent neural network based language model,” in Interspeech, 2010, pp. 1045–1048.

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, “Swin transformer: Hierarchical vision transformer using shifted windows,” in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10012–10022.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2018.

[8] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhudinov, “Transformer-xl: Attentive language modeling beyond a fixed-length context,” in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 2978–2987.

[9] Shigeki Karita, Naoxun Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyi Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, XiaoFei Wang, et al., “A comparative study on transformer vs rnn in speech applications,” in 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). IEEE, 2019, pp. 449–456.

[10] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al., “Conformer: Convolution-augmented transformer for speech recognition,” Interspeech, pp. 5036–5040, 2020.

[11] Yuan Gong, Yu-An Chung, and James Glass, “Ssast: Self-supervised audio spectrogram transformers,” arXiv preprint arXiv:2111.06377, 2021.

[12] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick, “Masked autoencoders are scalable vision learners,” arXiv preprint arXiv:2111.06377, 2021.

[13] Helin Wang, Yuexian Zou, Dading Chong, and Wenwu Wang, “Environmental sound classification with parallel temporal-spectral attention,” Interspeech, pp. 821–825, 2020.

[14] Yuan Gong, Yu-An Chung, and James Glass, “Psl: Improving audio event classification with pretrained, sampling, labeling, and aggregation,” arXiv e-prints, pp. arXiv–2102, 2021.

[15] Luya Wang and Aaron van den Oord, “Multi-format contrastive learning of audio representations,” arXiv preprint arXiv:2103.06508, 2021.

[16] Ilya Loshchilov and Frank Hutter, “Decoupled weight decay regularization,” arXiv preprint arXiv:1711.05101, 2017.

[17] Ilya Loshchilov and Frank Hutter, “Sgdr: Stochastic gradient descent with warm restarts,” arXiv preprint arXiv:1608.03983, 2016.

[18] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz, “mixup: Beyond empirical risk minimization,” arXiv preprint arXiv:1710.09412, 2017.

[19] Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning, “Electra: Pre-training text encoders as discriminators rather than generators,” arXiv preprint arXiv:2003.10550, 2020.

[20] Karol J Piczak, “Esc: Dataset for environmental sound classification,” in Proceedings of the 23rd ACM international conference on Multimedia, 2015, pp. 1015–1018.

[21] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen, “Acoustic scene classification in dcase 2019 challenge: Closed and open set classification and data mismatch setups,” 2019.

[22] Eric Humphrey, Simon Durand, and Brian McFee, “Openmic-2018: An open data-set for multiple instrument recognition,” in ISMIR, 2018, pp. 438–444.

[23] Pete Warden, “Speech commands: A dataset for limited-vocabulary speech recognition,” arXiv preprint arXiv:1804.03209, 2018.