CP-JKU SUBMISSION TO DCASE22: DISTILLING KNOWLEDGE FOR LOW-COMPLEXITY CONVOLUTIONAL NEURAL NETWORKS FROM A PATCHOUT AUDIO TRANSFORMER

Technical Report

Florian Schmid[1,2], *Shahed Masoudian*[2], *Khaled Koutini*[2], *Gerhard Widmer*[1,2]

[1]Institute of Computational Perception (CP-JKU),[2]LIT Artificial Intelligence Lab,
Johannes Kepler Universiy Linz, Austria
florian.schmid@jku.at, shahed.masoudian@jku.at, khaled.koutini@jku.at

ABSTRACT

In this technical report, we describe the CP-JKU team's submission for Task 1 *Low-Complexity Acoustic Scene Classification* of the DCASE 22 challenge [1]. We use Knowledge Distillation to teach low-complexity CNN student models from Patchout Spectrogram Transformer (PaSST) models. We use the pre-trained *PaSST* models on Audioset and fine-tune them on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset*. We experiment with using an ensemble of teachers, different receptive fields of the student models, and mixing frequency-wise statistics of spectrograms to enhance generalization to unseen devices. Finally, the student models are quantized in order to perform inference computations using 8 bit integers, simulating the low-complexity constraints of edge devices[1].

***Index Terms*—** Knowledge Distillation, Patchout Audio Transformer (PaSST), Receptive Field Regularization, Post-Training Static Quantization

1. INTRODUCTION

The first task of the DCASE'22 challenge is to recognize different scenes from one-second audio recordings [1]. This Acoustic Scene Classification (ASC) scenario presents various challenges and constraints:

- generalization across different recording devices and cities.
- low-complexity requirements in terms of number of parameters (128K), inference using 8 bit integer (INT8) computations, and a limited number of multiply-accumulate operations (30 million MACs);
- the limited amount of information contained in 1-second audio snippets compared to 10-second snippets in the ASC task of the DCASE'21 challenge [2].

Convolutional Neural Networks (CNNs) are well established models to tackle ASC tasks and have dominated previous years' challenges [2–5]. More specifically, Receptive Field Regularization (RFR) [6, 7] was shown to improve the generalization on this task [3, 8–10]. Recently, also Audio Transformers, like Patchout Audio Transformer (PaSST) [11], showed promising results on ASC tasks, achieving state-of-the-art accuracies on the *TAU Urban Acoustic Scenes 2020 Mobile dataset* [12]. PaSST models do not scale to the complexity requirements imposed by the challenge. This technical report describes our attempt to marry the concepts of PaSST models and RFR-CNNs in a Knowledge Distillation (KD) [13–15] framework. The ultimate goal is to compress the knowledge of a well-performing ensemble of PaSST models into a low-complexity RFR-CNN while maintaining the predictive performance. We perform Post-Training Static Quantization to convert weights and all computations of the final low-complexity model to INT8 variable type. In the following, we describe the techniques used to process the acoustic signals, train teacher and student models, and unify all parts in a teacher-student KD setup.

PaSST [11] and Audioset [16] are the only external data sources used in our submissions.

[1]Source Code: `https://github.com/CPJKU/cpjku_dcase22`

2. DATA PREPROCESSING AND AUGMENTATION

2.1. Preprocessing

The raw audio signal is down-sampled using a sampling rate of 32 kHz, and the input features are extracted from the raw audio signals using a Short Time Fourier Transformation (STFT) with a window size of 2048 and an overlap of 36% for the RFR-CNN student, and a window size of 800 with 40% overlap for the PaSST teacher models. We apply a Mel-scaled filter bank to end up with 256 frequency bins in case of the student and 128 frequency bins for the teacher models. The applied feature extraction for PaSST matches its Audioset pre-training settings and the student model feature extraction is similar to [7].

While training PaSST models downstream on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset*, we apply frequency and time masks of sizes 48 and 20, respectively. Additionally, pitch-shifting is applied by randomly changing the maximum frequency of the mel filter bank [17].

2.2. Reassembling 10-second Audio Files

The audio files are given in the form of 1-second snippets, which are obtained by splitting the full 10-second snippets of the *TAU Urban Acoustic Scenes 2020 Mobile development dataset* into 10 pieces. The provided meta-information includes segment identifiers that allow for reassembling the full 10-second recordings. To increase the diversity in the training data, we train our models by randomly cropping 1-second audio snippets from the reassembled 10-second version.

PaSST Settings for downstream training	**Accuracy**	**Log Loss**
$f = 0, p = 0.0$, *Mixup* ($\alpha = 0.3$)	0.596	1.304
$f = 4, p = 0.0$, *Mixup* ($\alpha = 0.3$)	0.604	1.119
$f = 8, p = 0.0$, *Mixup* ($\alpha = 0.3$)	0.600	1.063
$f = 8, \alpha = 0.8, p = 0.5$	0.602	1.064
$f = 6, \alpha = 0.8, p = 0.5$	0.612	1.050
$f = 4, \alpha = 0.8, p = 0.5$	0.612	1.077
$f = 2, \alpha = 0.8, p = 0.5$	0.606	1.123
$f = 6, \alpha = 0.8, p = 0.3$	0.612	1.054
$f = 6, \alpha = 0.8, p = 0.7$	0.603	1.066
$f = 6, \alpha = 0.6, p = 0.5$	0.611	1.050
$f = 6, \alpha = 0.4, p = 0.5$	0.611	**1.044**
PaSST_ENSEMBLE	0.627	1.001

Table 1: Results of PaSST models trained downstream on *TAU Urban Acoustic Scenes 2022 Mobile development dataset* on the provided development set split. f denotes frequency patchout, α denotes the mixing coefficient of *MixStyle*, p denotes the probability of a batch of recordings being selected for *MixStyle* and *Mixup* denotes the application of *Mixup* with a mixing coefficient of 0.3. Results are averaged over three different runs of the same setting. PaSST_ENSEMBLE uses averaged logits of eight models trained according to the different configurations that include *MixStyle* from the rows above.

2.3. Mixup and Mixstyle

We experiment with mixing feature and label information using *Mixup* [18] and mixing the style of the recordings using *MixStyle* [19]. *Mixup* has been shown to improve generalization in general [8] and *MixStyle* is used in particular to enhance generalization across recording devices. We experiment with a derivative of the original channel-wise *MixStyle* and apply *Freq-MixStyle*, as suggested in [20]. They show that the device-style is transported in the frequency-wise statistics (mean and standard deviation) of audio recordings. *Freq-MixStyle* is applied by normalizing frequency bands of spectrograms and denormalizing with mixed frequency components of two different recordings, where the mixing coefficient α specifies the shape of the Beta distribution. The effect of *Freq-MixStyle* can be regularized by a second parameter p, which controls the probability of whether *Freq-MixStyle* is applied to a batch of recordings. We applied *Mixup*, *Freq-MixStyle* and a combination of both and found that *Freq-MixStyle* outperforms *Mixup* and a combination across most settings. For both teacher and student models, we apply *Freq-MixStyle* only to the input.

3. TEACHER MODEL: PASST

With the rise of Vision Transformers (ViT) [21], processing spectrograms with transformer models became a recent topic of interest. Gong et al. [22] trained an Audio Spectrogram Transformer (AST) from pre-trained computer vision models to achieve state-of-the-art performance on Audioset [16]. Patchout faSt Spectrogram Transformer (PaSST) [11] extends this idea further by introducing a technique called *patchout* to tackle the quadratic scaling of attention layers with respect to the sequence length and to improve the generalization of trained transformers. PaSST models are well suited for training on downstream tasks in a short amount of time resulting, for instance, in an accuracy of 76.3% on the *TAU Urban Acoustic Scenes 2020 Mobile dataset* [12].

Spectrogram transformers [21–23] first extract overlapping patches of spectrograms and then add positional encodings to these patches. PaSST *disentangles* time and frequency positional encodings. This allows simple fine tuning of the pre-trained PaSST models on the shorter 1-second clips of the task [1]. For our experiments, we choose a PaSST model pre-trained on Audioset [16] with a patch size of 16x16 and a stride of 10. Structured patchout [11] is applied only on the *frequency* dimension, which removes entire frequency bins and shortens the resulting sequence of embeddings.

3.1. Experimental Setup

We train all PaSST models downstream on the *TAU Urban Acoustic Scenes 2022 Mobile development dataset* for a total of 250 epochs, where the model only sees one-tenth of the available data in a single epoch since we randomly crop a single 1-second piece for each 10-second recording. We use Adam [24] with a weight decay of 0.001 and a learning rate schedule: for the first 30 epochs the learning rate is exponentially increased to $1x10^{-5}$, followed by a linear decrease to $1x10^{-7}$ until epoch 130.

Table 1 shows the results on the provided development set split. f denotes the number of frequency bins to remove, and α and p denote the *MixStyle* configurations. The first row lists a model trained without patchout and *MixStyle*, which quickly overfits on the downstream task. Using patchout and *MixStyle* serves as proper regularization, with the removal of 6 frequency bins at random and moderate *MixStyle* configuration ($\alpha = 0.4$, $p = 0.5$) achieving the best results in terms of validation loss. Averaging the logits of multiple PaSST models of different configurations (denoted as PaSST_ENSEMBLE) further improves the results and suggests the use of transformer ensembles as teacher models.

4. STUDENT MODEL: RFR-CNN

The student model is a Receptive Field Regularized [6, 7] Convolutional Neural Network (RFR-CNN) and is based on *CP_ResNet* which performed well in previous editions of the DCASE challenge [2,5,8,12,17]. Table 2 summarizes the used CNN. We modify the architecture used in [17] in order to fit to the complexity constraints and to account for the shorter audio length in the DCASE 2022 challenge, as follows:

- We reduce the CNN's initial width to 32 channels. Since the number of parameters in the network grows quadratically with the width of the network, this greatly reduces the number of parameters.
- We introduce grouping in the convolutional layers of the network, which also reduces the number of the parameters and MACs regardless of their width [17]. This is controlled with the hyper-parameters G_2, G_3.
- We further fine-tune the number of parameters by changing the width of the final residual block of the network (the block with the majority of the parameters). We accomplish this using the hyper-parameter C.
- To account for the shorter audio clips (1-second), we change the max-pooling layers (indicated by P_f) to perform pooling only on the frequency dimension.
- We investigate two receptive field settings of the network, controlled via the hyper-parameter x.

Table 2: Compact CP-ResNet Architecture

WIDTH	GROUPING	BLOCK	CONFIG
W		INPUT	5×5, P
W	1	R	3×3, 1×1, P_f
W	1	R	3×3, 3×3, P_f
		LINEAR	$W \rightarrow 2\dot{W}$
$2 \times W$	G_2	R	$3 \times 3, 3 \times 3$
		LINEAR	$2W \rightarrow 4\dot{W}$
$4 \times W - C$	G_3	R	$x \times x$, 1×1
CLASSIFIER $4 \times W - C \rightarrow 10$ CLASSES			
GLOBAL MEAN POOLING			

P: 2×2 MAX POOLING.
P_f: 2×1 MAX POOLING OVER THE FREQUENCY DIMENSION.
R: RESIDUAL, THE INPUT IS ADDED TO THE OUTPUT

KD with varying temperatures	**Accuracy**	**Log Loss**
Baseline	0.524	1.418
T=1	0.541	1.216
T=3	**0.567**	**1.146**
T=5	0.559	1.183
T=8	0.554	1.219
T=12	0.551	1.265

Table 3: Varying temperature of soft targets for a simple low-complexity baseline model similar to 2 and a single teacher. Results are reported on the provided development set split.

KD with varying λ	**Accuracy**	**Log Loss**
λ=10	0.555	1.231
λ=30	0.563	1.175
λ=50	**0.577**	**1.138**
λ=100	0.565	1.144
λ=200	0.562	1.160

Table 4: Varying λ for a simple low-complexity baseline model similar to 2, a single teacher and $T = 3$. Results are reported on the provided development set split.

5. PASST AND RFR-CNN IN THE KNOWLEDGE DISTILLATION FRAMEWORK

In this section, we bring together PaSST teacher models and a low-complexity RFR-CNN in the KD framework. The aim is to compress the knowledge of multiple large teacher models into a low-complexity student model. We use KD in its original form, as introduced in [13]. First, PaSST models with different configurations are trained as shown in Table 1. Second, we choose PaSST models of different configurations to form a well-performing ensemble, which we denote as *PaSST_ENSEMBLE* in the following. Thirdly, a student model, as described in Section 4, is trained using a weighted sum of the distillation and the hard label loss as shown in Equation 1.

$$L_{TOTAL} = L_{LABEL} + \lambda L_{DIST} \tag{1}$$

The distillation loss is based on matching the predictions of student and teacher using soft targets given in Equation 2 with z being the logits, q the soft targets and T the temperature that controls the softness of the probability distributions.

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)} \tag{2}$$

Using the same temperature T to generate teacher and student soft targets from the respective logits, the distillation loss is calculated as the KL-divergence $L_{DIST} = D_{KL}(q_{student}||q_{teacher})$. Since 1-second pieces are cropped randomly from the reassembled 10-second recordings, *PaSST_ENSEMBLE* generates the soft targets in an online fashion, while the student model is trained.

Tables 3 and 4 show the effect of varying the temperature of the soft targets and the weight of the distillation loss λ, respectively. Both tables are based on a simple CNN baseline model and a single teacher and serve as a proof of effectiveness of KD from a PaSST teacher model. While $\lambda = 50$ is used in our final submissions, we found that using $T = 1$ is beneficial when switching to teacher ensembles.

Additionally, we experiment with adding a second distillation loss based on the predictions of a teacher on the full reassembled 10-second recordings, which adds a third term to the calculated loss as shown below. L_{DIST_LONG} is included into one of our submissions as it improved results slightly. Further investigations have to be conducted to show if it yields a stable improvement across different settings.

$$L_{TOTAL} = L_{LABEL} + \lambda L_{DIST} + \lambda_{LONG} L_{DIST\_LONG} \tag{3}$$

Finally, we investigate the use of a larger dataset for knowledge distillation. In particular, in addition to the previously discussed losses, we use a batch sampled from Audioset for each batch sampled from the development set, and we minimize the loss between the student and teacher on the audioset samples.

$$L_{TOTAL} = L_{LABEL} + \lambda(L_{DIST} + L_{DIST\_AUDIOSET}) \tag{4}$$

5.1. Experimental Setup

Student models in the KD framework are trained for a total of 750 epochs, where the model only trains on one-tenth of the available data in a single epoch since we randomly crop a single 1-second piece for each 10-second recording. We use Adam [24] with a weight decay of 0.001 and a learning rate schedule: for the first 150 epochs the learning rate is exponentially increased to $1x10^{-3}$, followed by a linear decrease to $5x10^{-6}$ until epoch 650.

Freq-MixStyle is applied to both teacher and student inputs independently, which enforces the student to match the teacher predictions even under different device-styles and enhances cross-device generalization. We set $\alpha = 0.3$ and experiment with different probabilities p.

6. QUANTIZATION

We use Post-Training Static Quantization as implemented in PyTorch [25] to quantize all model parameters and perform all infer-

ID	W,G_2,G_3,C,x	Mixup	Mixstyle	T, λ, λ_{LONG}, AS	Parameter Count	MMACs	LLoss	Quant. LLoss
1	32,2,1,36,3	$\alpha = 0.3$	✗	1,50,3,✗	127,046	29.06	1.111	1.115
2	32,2,1,36,3	✗	$\alpha = 0.3, p = 0.6$	1,50,0,✗	127,046	29.06	1.095	1.110
3	32,1,1,0,1	✗	$\alpha = 0.3, p = 0.2$	3,50,0,✗	121,610	28.24	1.127	1.139
4	32,1,1,0,1	$\alpha = 0.3$	✗	1,50,0, ✓	121,610	28.24	1.140	1.163

Table 5: Model configurations submitted to the challenge. **W, G_2, G_3, C** and **x** configure the student architecture as denoted in Section 4 and Table 2. **Mixup** and **MixStyle** describe the respective configurations when training the student model. **T** and **λ** denote temperature and the weight of the distillation loss. **λ_{LONG}** denotes the weight of the distillation loss with respect to a 10-second teacher and **AS** denotes the use of Audioset for knowledge transfer. The last two columns show the log losses on the provided development set split. LLoss is averaged over the last 10 epochs of training and Quant. LLoss is the result achieved by the final quantized model.

ID	Device-wise log losses								
	A	**B**	**C**	**S1**	**S2**	**S3**	**S4**	**S5**	**S6**
1	**0.755**	**1.021**	**0.886**	**1.092**	1.195	1.085	1.320	1.251	1.440
2	0.801	1.087	0.965	1.119	1.171	**1.072**	**1.216**	**1.172**	**1.384**
3	0.838	1.071	0.995	1.164	1.210	1.115	1.265	1.204	1.390
4	0.758	1.048	0.923	1.145	**1.169**	1.119	1.389	1.318	1.604

Table 6: Results of the quantized models on the provided development set split in terms of validation loss on a per-device level.

ence computations with 8 bit integers. Several steps are involved in the quantization procedure:

- Fusing all Convolution, Batch Norm and ReLU combinations to improve the numerical accuracy. This decreases parameters and increases MACs slightly. The provided complexity calculation tool is applied after this step.
- Insert observers to collect statistics that are used to calculate *zero point* and *scale* of quantization. We use *'fbgemm'* observer implemented in PyTorch.
- Collecting activation statistics using a calibration set. We use a subset of the training data for calibration.
- Quantizing all model parameters and the input data using the quantization stub inserted into the model's forward pass.

7. SUBMISSIONS AND RESULTS

The configurations and the final results on the development set split are reported in Table 5. Table 6 shows the loss on a per-device basis and underlines the performance gains of submissions including *Freq-MixStyle* (2 and 3) on the unseen devices **S4-S6**. Below we describe our submissions in detail:

Submission 1 (t10sec): This submission uses a student model with an initial width of 32 channels and an extended receptive field configured by setting **x=3** as listed in Table 2. To fulfill the complexity requirements, we use grouping of 2 on the penultimate residual block and decrease the width of the final residual block by setting **C=36**. We use *Mixup* with a mixing coefficient of 0.3. KD settings include a temperature of 1 and a distillation loss weight of 50. This submission uses a second distillation loss from a 10-second teacher with a weight of 3 as shown in Equation 3.

Submission 2 (mixstyleR8): This submission uses a student model configured in the same way as **Submission 1**. Instead of *Mixup*, this submission uses *MixStyle* with a mixing coefficient $\alpha =$ 0.3 and a probability of 0.6 to be applied to a certain batch. KD is performed using a temperature of 1 and a distillation loss weight of 50.

Submission 3 (mixstyleR5): This submission uses a student model with an initial width of 32 channels and a smaller receptive field configured by setting **x=1** as listed in Table 2. No additional grouping or decreasing width is necessary to stick to the complexity requirements. We use *MixStyle* with a mixing coefficient $\alpha = 0.3$ and a probability of 0.2 of being applied to a batch. KD settings include a temperature of 3 and a distillation loss weight of 50.

Submission 4 (audiosetR5): This submission uses a student model configured the same way as in **Submission 3**. We use *Mixup* with a mixing coefficient of 0.3. and a probability of 0.2 of being applied to a batch. KD settings include a temperature of 1 and a distillation loss weight of 50. This submission includes KD on Audioset as desribed in Equation 4.

All submitted models were retrained on the full development set before creating the final predictions for the challenge.

8. CONCLUSION

In this technical report, we described the CPJKU submission to Task 1 of the DCASE 22 challenge. We used ensembles of the audio spectrogram transformer PaSST to achieve a loss as low as possible. We then tried to compress the knowledge into a low-complexity receptive-field regularized CNN student model, while maintaining as much of the predictive performance as possible. We investigated Audioset as additional dataset for knowledge transfer and included the predictions of a 10-second teacher as distillation loss. Additionally, we experiment with *Freq-MixStyle* of spectrograms to enhance device generalization and, finally, restricted our model to perform all inference computations in variable type int8 using Post-Training Static Quantization.

9. ACKNOWLEDGMENT

The LIT AI Lab is financed by the Federal State of Upper Austria.

10. REFERENCES

[1] I. Martín-Morató, F. Paissan, A. Ancilotto, T. Heittola, A. Mesaros, E. Farella, A. Brutti, and T. Virtanen, “Low-

complexity acoustic scene classification in DCASE 2022 Challenge," 2022.

[2] I. Martín-Morató, T. Heittola, A. Mesaros, and T. Virtanen, "Low-complexity acoustic scene classification for multi-device audio: analysis of DCASE 2021 Challenge systems," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2021 Workshop (DCASE2021)*, 2021.

[3] B. Kim, S. Yang, J. Kim, and S. Chang, "QTI Submission to DCASE 2021: Residual Normalization for Device-Imbalanced Acoustic Scene Classification with Efficient Design," DCASE2021 Challenge, Tech. Rep., June 2021.

[4] C.-H. H. Yang, H. Hu, S. M. Siniscalchi, Q. Wang, W. Yuyang, X. Xia, Y. Zhao, Y. Wu, Y. Wang, J. Du, and C.-H. Lee, "A lottery ticket hypothesis framework for low-complexity device-robust neural acoustic scene classification," DCASE2021 Challenge, Tech. Rep., June 2021.

[5] K. Koutini, F. Henkel, H. Eghbal-zadeh, and G. Widmer, "CP-JKU Submissions to DCASE'20: Low-Complexity Cross-Device Acoustic Scene Classification with RF-Regularized CNNs," DCASE2020 Challenge, Tech. Rep., 2020.

[6] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "Receptive field regularization techniques for audio classification and tagging with deep convolutional neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 1987–2000, 2021.

[7] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The Receptive Field as a Regularizer in Deep Convolutional Neural Networks for Acoustic Scene Classification," in *Proceedings of the European Signal Processing Conference (EUSIPCO)*, A Coruña, Spain, 2019.

[8] K. Koutini, H. Eghbal-zadeh, and G. Widmer, "CP-JKU submissions to DCASE'19: Acoustic scene classification and audio tagging with receptive-field-regularized CNNs," DCASE2019 Challenge, Tech. Rep., June 2019.

[9] S. Suh, S. Park, Y. Jeong, and T. Lee, "Designing Acoustic Scene Classification Models with CNN Variants," DCASE2020 Challenge, Tech. Rep., 2020.

[10] K. Koutini, F. Henkel, H. Eghbal-Zadeh, and G. Widmer, "Low-complexity models for acoustic scene classification based on receptive field regularization and frequency damping," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, Tokyo, Japan, November 2020, pp. 86–90.

[11] K. Koutini, J. Schlüter, H. Eghbal-zadeh, and G. Widmer, "Efficient training of audio transformers with patchout," *CoRR*, vol. abs/2110.05069, 2021.

[12] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in DCASE 2020 Challenge: generalization across devices and low complexity solutions," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2020 Workshop (DCASE2020)*, 2020.

[13] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015.

[14] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 2014, pp. 2654–2662.

[15] C. Bucila, R. Caruana, and A. Niculescu-Mizil, "Model compression," in *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006*. ACM, 2006, pp. 535–541.

[16] J. F. Gemmeke, D. P. W. 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 *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, 2017.

[17] K. Koutini, S. Jan, and G. Widmer, "CPJKU Submission to DCASE21: Cross-Device Audio Scene Classification with Wide Sparse Frequency-Damped CNNs," DCASE2021 Challenge, Tech. Rep., June 2021.

[18] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[19] K. Zhou, Y. Yang, Y. Qiao, and T. Xiang, "Domain generalization with mixstyle," in *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[20] B. Kim, S. Yang, J. Kim, H. Park, J.-T. Lee, and S. Chang, "Towards robust domain generalization in 2d neural audio processing," 2021.

[21] 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 *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[22] Y. Gong, Y. Chung, and J. R. Glass, "AST: audio spectrogram transformer," in *Interspeech 2021, 22nd Annual Conference of the International Speech Communication Association, Brno, Czechia, 30 August - 3 September 2021*. ISCA, 2021, pp. 571–575.

[23] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, "Training data-efficient image transformers & distillation through attention," in *ICML 2021, 18-24 July 2021, Virtual*, vol. 139, 2021, pp. 10 347–10 357.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Z. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, 2019, pp. 8024–8035.