Enhancing Multiscale Features for Efficient Acoustic Scene Classification with One-Dimensional Separate CNN

Yuxuan He[1], *Alexander Raake*[2], *Jakob Abeßer*[3,4]

[1]Technische Universität Ilmenau, Germany [2]RWTH Aachen University, Germany
[3]University of Bamberg, Germany [4]Fraunhofer IDMT, Ilmenau, Germany

***Abstract*—Acoustic Scene Classification (ASC) is a fundamental task in audio signal processing, aiming to classify the location of an environmental audio recording. Recent advances focus on improving ASC model efficiency, particularly in resource-constrained environments. Convolutional neural networks (CNNs) remain the dominant approach due to their high performance, with recent focus on 1D kernels, such as in the Time-Frequency Separate Network (TF-SepNet), for reducing model complexity and computational cost. However, TF-SepNet performs feature extraction using a fixed receptive field in both time and frequency dimensions, which restricts its ability to capture multiscale contextual patterns. In this study, we investigate the integration of multiscale feature extraction modules into TF-SepNet, with the aim of improving model efficiency by balancing accuracy and complexity. We propose three architectures, TFSepDCD-Net, TFSepSPP-Net, and TFSepASPP-Net, each with two architectural variants based on TF-SepNet: one replaces its max pooling layers, and the other replaces its final convolutional layer. Each architecture has three configurations corresponding to different model sizes—small, medium, and large—to explore the tradeoff between accuracy and model complexity. Our experiments show that incorporating multiscale modules allows smaller models to achieve comparable or even superior accuracy to larger baselines. These findings highlight the potential of multiscale representations for improving the efficiency of CNN-based ASC systems, especially in 1D separate architectures like TF-SepNet.**

***Index Terms*—Acoustic scene classification, time-frequency separate network, 1D kernels, multiscale feature extraction**

1. INTRODUCTION

Acoustic scene classification (ASC) [1] is a fundamental task in audio signal processing, where the goal is to classify audio recordings into specific predefined environmental sound scenes such as shopping mall, metro, or office. Real-world ASC applications demand efficient operation under limited resources [2]. However, high-performing ASC models are often complex and resource-intensive. Thus, achieving a balance between accuracy and efficiency remains a key challenge [3].

Most ASC models build on convolutional neural networks (CNNs) [4]. Researchers have explored CNNs to improve both accuracy and efficiency in ASC tasks [2]. Moreover, the focus in recent studies has changed from traditional two-dimensional (2D) convolutional kernels to one-dimensional (1D) kernels to reduce computational complexity [5]. Unlike 2D kernels that process both time and frequency dimensions simultaneously, 1D kernels separate these processes, allowing for a significant reduction in the overall number of parameters [6]. This approach not only preserves the essential features of the audio signal, but also enhances the ability of the model to capture relevant time-frequency patterns by expanding the effective receptive field (ERF), which refers to the range of input regions that influence a model's prediction [7]. The use of 1D kernels in architectures like Time-Frequency Separate Network (TF-SepNet) [5] represents an important advancement in balancing efficiency and accuracy in ASC tasks. The design of TF-SepNet is based on the BC-ResNet architecture [8], which integrates depthwise separable convolutions [7] and broadcasting operations [8] to further reduce model complexity. Using separate convolutional paths for time and frequency features, the network effectively expands the effective ERF with fewer layers.

Although TF-SepNet successfully balances the ERF and model complexity, its fixed scale of feature extraction may limit its ability to fully capture the acoustic variations present in complex acoustic environments. A promising strategy to overcome this limitation is to incorporate multiscale representations, which enable models to analyze features at different resolutions, ensuring that both details and broader contextual patterns are effectively captured [9], [10]. One such technique is Dilation Convolution Downsampling (DCD), which utilizes dilated convolutions with exponentially increasing dilation rates and varying kernel sizes to downsample time series data at multiple scales [11]. This enables the network to extract complex temporal features by capturing both short-term local dependencies as well as long-term global dependencies [12]. Another technique is Spatial Pyramid Pooling (SPP) [13], which has been used in computer vision to pool features at multiple spatial scales. In the context of ASC, SPP has demonstrated its potential to improve classification accuracy by aggregating local features of convolutional layers on different scales [14]. Building on the concept of SPP, Atrous Spatial Pyramid Pooling (ASPP) [15] enhances multiscale feature extraction by integrating atrous (dilated) convolutions into the pyramid pooling framework [16].

As the main contribution of this paper, we propose three novel architectures named **TFSepDCD-Net**, **TFSepSPP-Net**, and **TFSepASPP-Net**, which integrate DCD, SPP, and ASPP into different components of TF-SepNet, respectively. Each architecture has two variants and three sub-models with channel widths of 60, 40, and 20, using the original TF-SepNet [5] [17] as the baseline. Our goal is to explore how multiscale representations influence the balance between model complexity and performance, aiming to find configurations that reduce parameters and computation while maintaining accuracy. We hypothesize that introducing lightweight multiscale modules into TF-SepNet enables a better trade-off between classification accuracy and model complexity.

2. METHODOLOGY

2.1. Time-Frequency Separate Network (TF-SepNet)

TF-SepNet [5] is a CNN architecture specifically designed for ASC tasks, aiming to achieve a balance between model complexity and classification accuracy. The core of TF-SepNet lies in the TF-SepConvs module, as shown in Fig.2 of [5]. This module is specifically designed to independently process the time and frequency dimensions of input features using 1D convolutional kernels. The TF-SepConv module begins with a transition layer, implemented as a 1×1 convolution, which adjusts the number of channels. Following this, a shuffle unit is applied to facilitate the mixing of information across channels before splitting the feature maps into two separate paths: the frequency path and the temporal path. In the frequency path, a depthwise convolution with a kernel size of 3×1 is applied, followed by frequency average pooling. Similarly, in the temporal path, a depthwise convolution with a kernel size of 1×3 is followed by temporal average pooling.

Output Shape	Architecture	k	s	p
1, F, T	Input	-	-	-
C/2, F/2, T/2	ConvBnRelu	5	2	2
2C, F/4, T/4	ConvBnRelu	1	1	0
C, F/4, T/4	TF-SepConvs ×2	-	-	-
C, F/8, T/8	Max Pooling	2	2	0
1.5C, F/8, T/8	TF-SepConvs ×2	-	-	-
1.5C, F/16, T/16	Max Pooling	2	2	0
2C, F/16, T/16	TF-SepConvs ×2	-	-	-
2.5C, F/16, T/16	TF-SepConvs ×3	-	-	-
10, F/16, T/16	Conv	1	1	0
10, 1, 1	Average Pooling	-	-	-

Table 1: **TF-SepNet network architecture** details showing the output shape, layers, and their respective kernel size (k), stride (s), and padding (p).

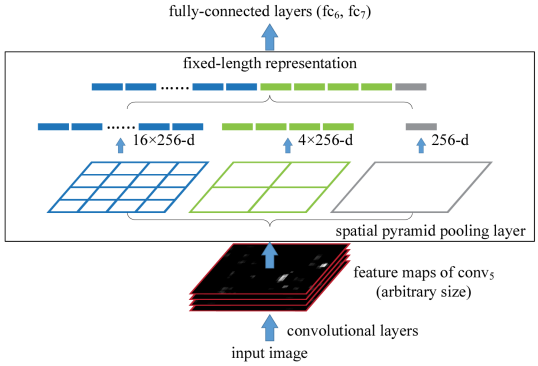

Fig. 1: **A CNN structure with an SPP layer**, as cited in [18].

Both paths then employ a pointwise convolution to further process the features. The output features from the frequency path and the temporal path are then broadcasted back to their original dimensions and concatenated along the channel axis to form the final output.

The architecture of TF-SepNet, as outlined in Table 1, begins with two initial 3×3 convolution layers with a stride of 2, which perform early downsampling of the input spectrogram. This is followed by a series of nine TF-SepConv modules interspersed with two 2×2 max-pooling layers that further reduce spatial dimensions while capturing higher-level features. In the final stage, a 1×1 convolutional layer is used, followed by global average pooling to generate multiclass probabilities as the model's output. An adaptive residual normalization technique is integrated after the initial downsampling block and after each TF-SepConvs block to ensure stability and to improve convergence during training [17]. A key parameter of the TF-SepNet model is its channel width, denoted by the hyperparameter τ, which allows to adjust the model complexity. By tuning τ, TF-SepNet can be scaled to meet different computational requirements, from resource-limited environments to high-performance computing systems.

2.2. Dilation Convolution Downsampling (DCD)

DCD is a technique that captures both short-term and long-term dependencies in time series data by applying dilated convolutions with exponentially increasing dilation rates [11]. This approach expands the receptive field without increasing the number of parameters, allowing the model to efficiently extract hierarchical temporal features across different resolutions. Moreover, DCD has been successfully applied in various tasks. For example, in [12], the authors employ a Multi-Scale Dilated Convolution Network (MSDCN) with dilation rates of 1, 2, and 4, and kernel sizes of 2, 4, and 6 to downsample and process time series data.

2.3. Spatial Pyramid Pooling (SPP)

SPP is a multiscale feature aggregation technique that addresses the limitation of fixed-size inputs in CNNs by introducing an SPP layer. For instance, as illustrated in Fig. 1, the SPP layer is inserted between the final convolutional layer and the fully connected layers. It pools local features over multiple spatial bins and captures hierarchical spatial information at different resolutions. Each spatial bin performs max pooling over regions of the convolutional feature maps, and the pooled outputs are flattened and concatenated. When the last conv layer outputs 256 feature channels, pooling over 4×4, 2×2, and 1×1 bins yields 16, 4, and 1 pooled values per channel, resulting in feature maps of shape 16×256, 4×256, and 256. These are concatenated into a single fixed-length vector that feeds into the fully connected layers. This design enables the network to handle inputs of arbitrary sizes while preserving multiscale spatial structure in the feature representation [18].

2.4. Atrous Spatial Pyramid Pooling (ASPP)

ASPP [15] is a powerful technique designed to capture multiscale contextual information [10] by applying parallel atrous (dilated) convolutions with varying dilation rates. The ASPP module consists of multiple convolutional layers, typically with a kernel size of 3×3, and different dilation rates commonly set to 6, 12, 18, and 24. These dilation rates allow the module to capture features at different scales, effectively covering various receptive fields. In addition to these dilated convolutions, ASPP also includes a global pooling branch, which provides global contextual information by pooling the entire feature map into a single value per channel. The outputs from the dilated convolutions and a global pooling branch are then concatenated along the channel dimension, resulting in a multiscale feature map. This map is passed through a 1×1 convolution to reduce the number of channels, integrating the multiscale information into a compact representation.

2.5. The Proposed Architectures

As the main contribution of this paper, we propose three novel architectures—**TFSepDCD-Net**, **TFSepSPP-Net**, and **TFSepASPP-Net**—which integrate DCD, SPP, and ASPP into different components of TF-SepNet. For each architecture, we develop two variants: (1) the two max pooling layers originally used for intermediate downsampling in TF-SepNet are replaced with the respective modules (**MP variants**); (2) the final convolutional layer originally used in TF-SepNet is replaced with the respective module (**LP variants**).

2.5.1. TFSepDCD-Net: In the first architecture, which we refer to as TFSepDCD-MP, the two max pooling layers originally used for intermediate downsampling in TF-SepNet are replaced with DCD modules. Each DCD module begins with a 3×3 convolutional layer with stride 2 to perform spatial downsampling, reducing the input resolution from $C \times F/8 \times T/8$ to $C \times F/16 \times T/16$. The downsampled feature map is then processed through multiple parallel dilated convolutions with increasing dilation rates to capture contextual information at different temporal and spectral resolutions. Each dilated convolution uses a 3×3 kernel with the dilation rates $d \in \{1, 2, 3\}$ and produces 8-channel outputs. The operation of the DCD module can be defined as:

$$v_{dcd} = \sum_{d \in \{1,2,3\}} D_d(x'), \tag{1}$$

where x' is the downsampled input feature map, $D_d(\cdot)$ denotes a 3×3 convolution with dilation d. These outputs are concatenated

and passed through a final 1×1 convolution to maintain spatial consistency and reduce the feature map to the desired number of channels.

In the second architecture, TFSepDCD-CL, the final convolutional layer before global average pooling in TF-SepNet is replaced with a DCD module. The multiscale dilated structure mirrors that of TFSepDCD-MP, using dilation rates of 1, 2, and 3. However, unlike TFSepDCD-MP, this module does not perform downsampling. Instead, the input feature map is first passed through a 1×1 convolution to reduce the number of channels from $2.5 \cdot C$ to C, after which the DCD branches are applied.

2.5.2. TFSepSPP-Net: In the first architecture, which we refer to as TFSepSPP-MP, the two max pooling layers originally used for intermediate downsampling in TF-SepNet are replaced with SPP modules. As illustrated in Figure 2, the process begins by applying a convolutional layer with a stride of 2 to downsample the input feature map to the same resolution it would have after max pooling. The downsampled feature map is then processed through the SPP module, which captures multiscale information using three Adaptive Average Pooling (AAP) branches with output sizes of 1×1, 2×2, and 4×4, along with a global pooling branch. Each branch includes a 1×1 convolution to process the pooled feature into a lower-dimensional representation, which is then upsampled to the original resolution. The resulting multiscale features are concatenated and passed through a final 1×1 convolution to preserve the spatial size and match the desired number of channels. The overall operation is defined as:

$$v_{spp} = \sum_{k \in \{1,2,4\}} W_k \cdot U\left(P_k(x')\right) + W_g \cdot G(x'), \tag{2}$$

where x' denotes the downsampled feature map, $P_k(\cdot)$ represents adaptive average pooling to a $k \times k$ grid, $U(\cdot)$ denotes bilinear upsampling, W_k and W_g are 1×1 convolution, and $G(x')$ is the global pooling output.

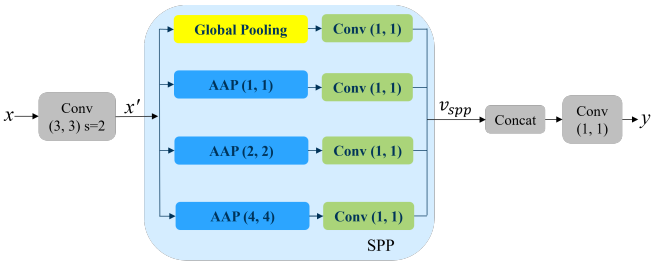

Fig. 2: Visualization of the application of the SPP module in the TFSepSPP-MP.

In the second architecture, TFSepSPP-CL, the final convolutional layer before global average pooling in TF-SepNet is replaced with an SPP module. As in the TFSepSPP-MP, the module includes three scale branches and a global pooling path. However, there is a difference: the input feature map is initially passed through a 1×1 convolution to reduce the number of channels from $2.5C$ to C, enabling more efficient multiscale processing.

2.5.3. TFSepASPP-Net: In the first architecture, which we refer to as TFSepASPP-MP, the two max pooling layers originally used for intermediate downsampling in TF-SepNet are replaced with ASPP modules. As illustrated in Figure 3, the process involves first applying a convolutional layer with a stride of 2. Then, the downsampled feature map is processed through the ASPP module, which consists of three dilated convolutions with dilation rates of 6, 12, and 18, and a global pooling branch. The global pooling branch applies average pooling across the entire feature map to capture global context information, followed by a 1×1 convolution denoted as $W_{1\times1}$ to match the number of channels with the other branches. The resulting multiscale features from all branches are then concatenated as shown in Equation 3, where x' represents the downsampled input feature map, W_d denotes the dilated convolution operations with zero padding matching the dilation rate, and $G(x')$ denotes the global pooling operation followed by the 1×1 convolution. Finally, the concatenated feature map is processed through a 1×1 convolution to maintain the original channel count and spatial dimensions.

$$v_{aspp} = \sum_{d \in \{6,12,18\}} W_d \cdot x' + W_{1\times1} \cdot G(x'). \tag{3}$$

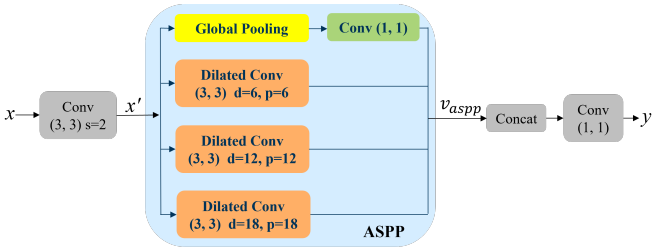

Fig. 3: Visualization of the application of the ASPP module in the TFSepASPP-MP.

In the second architecture, TFSepASPP-CL, the final convolutional layer before global average pooling in the original TF-SepNet is replaced by an ASPP module. The input feature map initially passes through a 1×1 convolution to adjust the channel dimensions before entering the ASPP module.

3. EVALUATION AND RESULTS

3.1. Dataset and Pre-processing

We use the TAU Urban Acoustic Scenes 2022 Mobile development dataset [19]. We follow the official 7:3 training/test split for our experiments. Our feature extraction pipeline strictly follows the original TF-SepNet implementation [17]: All audio segments are downsampled to 32 kHz and the time-frequency features are extracted using the Short-Time Fourier Transform (STFT) with a window size of 3072 samples (64 ms) and a hop size of 500 samples (10.42 ms). A Mel-scaled filter bank with 256 frequency bands and 4096 FFT points is used to convert the spectrograms into Log-Mel spectrograms. We also incorporate device simulation as done in [17].

3.2. Training Setup

The training setup strictly follows that of the original TF-SepNet [5] [17]. The model is trained for 100 epochs using the Adam optimizer. The batch size is 32. A warm-up strategy is also applied, where the learning rate is gradually increased from 0 to 0.01 during the initial five epochs, followed by a gradual reduction to 0 using cosine annealing for the remaining training epochs. Mixup [20] and Freq-MixStyle [21] techniques are incorporated, where Mixup's α is set to 0.3, and Freq-MixStyle's α and ρ values are set to 0.3 and 0.7.

3.3. Experimental Design

In our experiments, we evaluate a total of 21 models, including the original TF-SepNet [5] [17] as the baseline and three proposed architectures, each one being implemented as both **MP** and **LP** variant. All models are evaluated under three different configurations by

Model	Variant	τ	Params / k	MACs / M	Acc. / %
TFSepNet	–	60	115.15	15.20	66.2
		40	54.27	7.03	64.3
		20	15.89	3.42	58.5
TFSep-DCD	MP	60	217.90	44.42	65.3
		40	106.87	22.38	63.7
		20	34.24	7.58	63.3
	CL	60	107.59	21.49	66.8
		40	56.81	11.20	64.2
		20	21.03	4.04	64.1
TFSep-SPP	MP	60	196.15	38.79	61.9
		40	92.37	18.64	62.7
		20	26.99	5.70	64.2
	CL	60	90.31	18.93	66.4
		40	45.53	9.51	64.4
		20	15.75	3.23	63.7
TFSep-ASPP	MP	60	281.02	38.29	62.9
		40	128.15	17.96	62.9
		20	34.48	5.20	61.7
	CL	60	210.64	22.27	66.0
		40	96.63	10.76	63.0
		20	26.42	3.33	65.0

Table 2: Model accuracy along with parameter count and MACs consumption for TFSepNet and the proposed variants across different channel widths. Each proposed model includes two architectural variants (MP, CL).

varying the channel width $\tau \in \{60, 40, 20\}$, corresponding to large, medium, and small model sizes.

3.4. Results

As shown in Table 2, all proposed models demonstrate clear performance improvements over the TF-SepNet baseline using the smallest channel width of $\tau = 20$. Among the two architectural variants, the CL variant is more effective for lightweight configurations, consistently outperforming its MP counterparts. Models such as **TFSepDCD-CL20** (64.1%), **TFSepSPP-CL20** (63.7%), and **TFSepASPP-CL20** (65.0%) achieve higher accuracy compared to the baseline **TF-SepNet-20** (58.5%), with only marginal increases in model complexity. Notably, many of these small models perform on par with or better than TF-SepNet-40 (64.3%), which has more than twice the number of parameters. This shows that lightweight multiscale variants can reach comparable accuracy while substantially reducing the model size. Moreover, among all models, **TFSepSPP-CL20** stands out by offering an excellent trade-off: it achieves 63.7 % accuracy with just 15.75 k parameters and 3.23 M MACs, nearly matching the baseline's size while surpassing it in accuracy by 5.2%. These findings are visually summarized in Figure 4, which plots model accuracy against number of parameters. For clarity, only the baseline TF-SepNet models with $\tau = 20$ and 40, along with all proposed variants at $\tau = 20$, are highlighted.

To further assess the performance-efficiency tradeoff of our proposed architectures, we conducted significance tests on both accuracy and parameter count across different model groups, as summarized in Table 3. Apart from the comparison between all $\tau = 20$ models and all $\tau = 60$ models, which shows a significant accuracy difference ($p = 0.034$), all other comparisons yield $p > 0.05$ for accuracy. Notably, when combining all CL-20 and MP-20 variants and comparing them to all $\tau = 40$ models, the p-value reaches as high as 0.928, indicating virtually no performance difference between our smallest models and the larger $\tau = 40$ configurations. All comparisons in terms of parameter count show statistically significant differences ($p < 0.01$). Moreover, we also individually tested CL-20 and MP-20

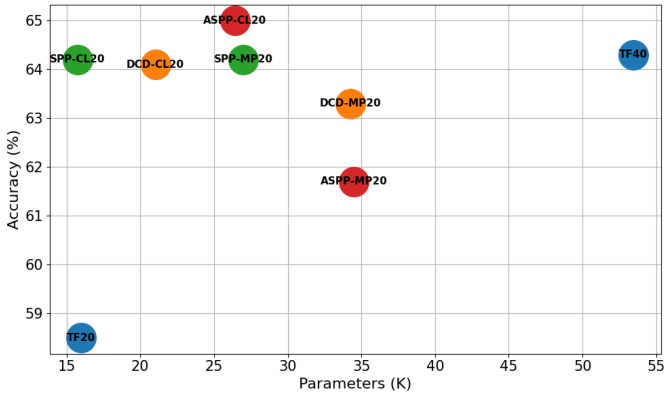

Fig. 4: **Accuracy versus parameter size for selected models.** Color coding indicates different model types: blue for the baseline TF-SepNet, orange for TFSepDCD-Net variants, green for TFSepSPP-Net variants, and red for TFSepASPP-Net variants.

Comparison	p-value (Accuracy)	p-value (Params)
All-20 vs All-40	0.257	**0.002**
All-20 vs All-60	**0.034**	$<$ **0.001**
CL-20&MP-20 vs All-40	0.928	**0.003**
CL-20&MP-20 vs All-60	0.068	$<$ **0.001**

Table 3: **Significance test results (p-values) for accuracy and parameter count.** Each row compares a group of lightweight models with mid-size or large counterparts. Bold values indicate statistically significant differences ($p < 0.05$).

against all $\tau = 40$ and $\tau = 60$ models. In all four comparisons, the differences in accuracy were not statistically significant ($p = 0.306$, 0.326, 0.346, and 0.084).

4. CONCLUSION

This paper presented three novel ASC model architectures, which integrate multiscale feature extraction modules into the efficient, 1D-kernel-based TF-SepNet backbone. Each architecture was implemented in two variants, replacing either the final convolutional layer or the max-pooling layers, and evaluated across three different channel configurations ($\tau = 20$, 40, 60). Our experiments demonstrated that the proposed models with $\tau = 20$ notably outperform the baseline TF-SepNet-20, achieving higher classification accuracy with only a marginal increase in parameter count. Among these, TFSepASPP-CL20 achieved a 65.0% accuracy, representing a 6.5% absolute improvement over TF-SepNet-20. Furthermore, statistical significance tests confirmed that, except for the direct comparison between all $\tau = 20$ models and all $\tau = 60$ models, the accuracy differences between the lightweight $\tau = 20$ variants and their larger $\tau = 40$ and $\tau = 60$ counterparts are not statistically significant ($p > 0.05$), with a particularly high p-value of 0.928 when comparing all CL- and MP-based $\tau = 20$ models with all $\tau = 40$ models. In contrast, all comparisons in terms of parameter count yielded statistically significant differences ($p < 0.01$), reinforcing that our proposed models are substantially more compact. These findings support our hypothesis, showing that multiscale variants of TF-SepNet achieve a more favorable balance between accuracy and efficiency, and in particular, the multiscale-integrated $\tau = 20$ variants highlight this trade-off by achieving competitive performance with substantially reduced model complexity.

REFERENCES

[1] D. Barchiesi, D. Giannoulis, D. Stowell, and M. D. Plumbley, "Acoustic scene classification: Classifying environments from the sounds they produce," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 16–34, 2015.

[2] B. Ding, T. Zhang, C. Wang, G. Liu, J. Liang, R. Hu, Y. Wu, and D. Guo, "Acoustic scene classification: A comprehensive survey," *Expert Systems with Applications*, vol. 238, p. 121902, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417423024041

[3] Y. Li, W. Cao, W. Xie, Q. Huang, W. Pang, and Q. He, "Low-complexity acoustic scene classification using data augmentation and lightweight resnet," in *2022 16th IEEE International Conference on Signal Processing (ICSP)*, vol. 1, 2022, pp. 41–45.

[4] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, 2015, pp. 1–6.

[5] Y. Cai, P. Zhang, and S. Li, "Tf-sepnet: An efficient 1d kernel design in cnns for low-complexity acoustic scene classification," in *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2024, pp. 821–825.

[6] D. H. Phan and D. L. Jones, "Low-complexity acoustic scene classification using time frequency separable convolution," *Electronics*, vol. 11, no. 17, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/17/2734

[7] W. Luo, Y. Li, R. Urtasun, and R. Zemel, "Understanding the effective receptive field in deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., vol. 29. Curran Associates, Inc., 2016. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2016/file/c8067ad1937f728f51288b3eb986afaa-Paper.pdf

[8] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted residual learning for efficient keyword spotting," in *Proceedings of the Conference of the International Speech Communication Association (INTERSPEECH)*. ISCA, 2021, pp. 4538–4542.

[9] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille, "Attention to scale: Scale-aware semantic image segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[10] I. Kokkinos, "Pushing the boundaries of boundary detection using deep learning," *arXiv: Computer Vision and Pattern Recognition*, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:15197911

[11] A. Borovykh, S. M. Bohte, and C. W. Oosterlee, "Dilated convolutional neural networks for time series forecasting," *Journal of Computational Finance*, vol. 22, pp. 73–101, 2017. [Online]. Available: https://ir.cwi.nl/pub/28485/28485.pdf

[12] F. Li, S. Guo, F. Han, J. Zhao, and F. Shen, "Multi-scale dilated convolution network for long-term time series forecasting," 2024. [Online]. Available: https://arxiv.org/abs/2405.05499

[13] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006, pp. 2169–2178.

[14] A. M. Basbug and M. Sert, "Acoustic scene classification using spatial pyramid pooling with convolutional neural networks," in *2019 IEEE 13th International Conference on Semantic Computing (ICSC)*, 2019, pp. 128–131.

[15] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[16] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[17] Y. Cai, M. Lin, C. Zhu, S. Li, and X. Shao, "Dcase2023 task1 submission: Device simulation and time-frequency separable convolution for acoustic scene classification," *Tech. Rep., Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge*, 2023.

[18] K. He, X. Zhang, S. Ren, and J. Sun, *Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition*. Springer International Publishing, 2014, p. 346–361. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10578-9_23

[19] T. Heittola, A. Mesaros, and T. Virtanen, "Tau urban acoustic scenes 2022 mobile, development dataset," *Zenodo*, 2022.

[20] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[21] F. Schmid, S. Masoudian, K. Koutini, and G. Widmer, "Cp-jku submission to dcase22: Distilling knowledge for low-complexity convolutional neural networks from a patchout audio transformer," *Tech. Rep., Detection and Classification of Acoustic Scenes and Events (DCASE) Challenge*, 2022.