Searching for EfficientNeXt

Luca Careddu*

***Abstract*—Audio Classification is an active research area in the Deep Learning field with a wide range of applications in both industry and everyday life. In the last years, we have seen the power of Large Language Models (LLMs) and the importance of pretraining as a learning baseline for models even when the transferred knowledge belongs to different domains. Training LLMs is by the way expensive due to the transformer architecture they are based on which is very data-demanding and pretraining on large datasets is often unfeasible so CNNs are still widely used when resources are limited. In this work, we classify audios from both FSD50K and ESC-50 datasets using deep neural networks based on a variation of the MobileNetV3 and the Fused inverted bottlenecks that highly reduce the complexity of the original ones, especially as the number of input channels grows, while increasing their expressive power by exploiting grouped convolutions and the Squeeze and Excitation mechanism applied both locally within the group and globally between the groups. Results from the experiments show that our approach drastically reduces the complexity of the baseline model, improves its prediction performances, and, using fewer FLOPS and less GPU memory, also exceeds the results of the state-of-the-art pure CNNs models EfficientNet-B2 and EfficientNetV2-B2 when these are not pretrained on ImageNet, in particular our final model achieves 0.488 mAP on FSD50K and 0.967 accuracy on ESC-50 without an ImageNet pretraining.**

***Index Terms*—Environmental Sound Classification, Convolutional Neural Networks, MobileNet, ResNeXt, EfficientNet.**

I. Introduction

Environmental Sound Classification (ESC) is a task in Deep Learning that aims to build models able to distinguish audios belonging to classes such as animal sounds, natural soundscapes and water sounds, human (non-speech) sounds, interior/domestic sounds, and exterior/urban noises. Unlike the Speech Recognition task, where audio samples can be represented as sequences of phonemes and these can be used to distinguish words, speaker accent, speaker sex, and so on, ESC samples typically lack structure and do not represent anything semantically meaningful such as a discourse or dialogue resulting in a more difficult classification task.

Up to the time of writing transformers-based models hold the state-of-the-art in most of the Natural Language Processing, Computer Vision, and also sound-related tasks (e.g. sound tagging, see [1], [2], and [3]). Despite their impressive power, these models are very expensive to train in terms of the huge data demand and computational resources required, and for these reasons, CNNs are still employed in the field. A common factor of models holding state-of-the-art performances is typically pretraining, which is usually needed as a priori knowledge and can be performed on data that do not share the same nature as the target data. In this regard, [4] was, to the best of our knowledge, the first study to prove the advantages of using pretraining on large image datasets when solving audio tasks. In the work, they show how images and spectrograms share common patterns which are typically those that are learned by the bottom layers of standard CNNs. Subsequent works proved that the same argument holds for transformer-based models. Pretraining promotes the reuse of already existing models but can be unfeasible for new models. Recently, remarkable results have been obtained transferring knowledge from transformers to CNNs through Knowledge Distillation ([5]) and from both directions through Cross-Model Knowledge Distillation ([6]).

In this work, we exploit the main ideas found in the studies of the MobileNet [7]–[9] and EfficientNet [10,11] series along with the result of the ResNeXt work [12], which by simply branching internal convolutions of the bottleneck into independent convolutions managed to increase the expressiveness of the network and, without increasing the complexity, improved its predecessor ResNet [13] classification performance, to build a lightweight model based on a variation of the MobileNetV3 [9] and the Fused [14] inverted bottlenecks that is able to decrease the complexity of the baseline one and exceed, using fewer FLOPS, its classification performances on both FSD50K [15] and ESC-50 [16] datasets, achieving 0.488 mAP on the former and 0.967 accuracy on the latter. To perform a meaningful comparison, we choose two pure CNNs state-of-the-art models from the EfficientNet series that roughly have the same number of parameters as our final model, namely EfficientNet-B2 and EfficientNetV2-B2, and compare our results with their on both datasets to show that the model following our approach, using fewer FLOPS and less GPU memory, exceeds their classification performance when they are not pretrained on ImageNet [17]. Our version of the inverted bottlenecks exploits grouped convolutions to reduce the complexity and FLOPS of the original ones, especially as the number of input channels increases while encouraging the expressive power through the use of the soft-attention Squeeze and Excitation mechanism [18] applied both locally between the channels of each group and globally between the channels of all groups. We finish the work by performing multiple ablation studies to get some insight into how our approach works under the hood and by finally drawing conclusions.

The results obtained in this work are easily reproducible with the code available at https://github.com/lucacareddu/Searching-for-EfficientNeXt in both PyTorch (used to perform the experiments) and TensorFlow versions and we believe that further improvements and experiments can be done, for example introducing a Neural Architecture Search to build possibly

*Department of Mathematics "Tullio Levi-Civita", University of Padua, email: luca.careddu@studenti.unipd.it

the best model exploiting our approach while remaining within a fixed budget of FLOPS and/or GPU memory, pretraining the model on a large dataset such as AudioSet [19] and ImageNet, improving the training phase with more data augmentations and techniques such as using an ensemble model, weights averaging, etc. To summarize, our contributions are:

- We introduce an approach that improves the MobileNetV3 and Fused inverted bottlenecks by exploiting grouped convolutions to significantly reduce complexity and FLOPS, and the Squeeze and Excitation mechanism applied both locally and globally to increase expressiveness.
- We propose a model that, based on our inverted bottlenecks, is able to achieve competitive mAP on FSD50K and nearly state-of-the-art accuracy on ESC-50.
- We interpret our results by comparing them with those of pure CNNs state-of-the-art models from the EfficientNet series to prove the efficiency of our method and investigate how our approach works under the hood by performing multiple ablation studies.

This paper is organized as follows. Section II presents the related work, Section III presents the datasets and discusses the data preprocessing pipelines, Section IV introduces the baseline model, Section V introduces the approach, Section VI discusses and interprets the results, and Section VII reports multiple ablation studies. Finally, Section VIII draws the conclusions and future work.

II. Related Work

A. Transformer-based models

Audio Spectrogram Transformer (AST) [20] is able to achieve state-of-the-art performance on AudioSet and ESC-50 by using the ImageNet-pretrained Vision Transformer (ViT) model [21] as the backbone. SSAST [22] boosts AST performance by training the model in a self-supervised manner. BEATs [3] exceeds state-of-the-art performance by exploiting both self-supervised learning (SSL) and knowledge distillation (KD). OmniVec [1] joins a composition of multiple task-related encoders with a transformer-based common branch and, trained on all major modalities (e.g. image, video, depth map, speech, and text), it achieves state-of-the-art results on most of the benchmarks.

B. Knowledge Distillation

In [5] transfer learning in the form of Knowledge Distillation from transformers to CNNs results in state-of-the-art mAP on AudioSet, while in [6] they show that transformers and CNNs can learn and gain from each other thanks to Cross-Model Knowledge Distillation.

C. CNNs

ResNeXt [12] introduces a multi-branch architecture that exposes a new dimension, which they call ”cardinality” (the size of the set of transformations), as an essential factor in addition to the dimensions of depth and width. In the work, they show that cardinality is more effective than going deeper or wider when capacity is increased. DenseNet [23] alleviates the vanishing-gradient problem, strengthens feature propagation, encourages feature reuse, and substantially reduces the number of parameters by feeding each layer with the feature maps of all preceding layers as inputs. The MobileNet series [7]–[9] introduces a novel block called Inverted Residual with Linear Bottleneck that factorizes the standard convolution into three components (Pointwise expansion convolution, Depthwise convolution, Pointwise linear convolution) and that requires a lower number of FLOPS and GPU memory to work. They also introduced a way to scale the baseline models using a width and resolution multiplier to obtain smaller and more efficient networks. MobileNetV3 [9] inserts inside the inverted bottleneck the Squeeze and Excitation soft-attention mechanism introduced in [18] to weight the channels without increasing noticeably the complexity, and has its baseline built using a Neural Architecture Search (NAS). The EfficientNetV1 [10] models, built upon the MobileNetV3 inverted bottleneck via NAS and a newly introduced compound coefficient that scales uniformly all three dimensions, achieves state-of-the-art performance in most image datasets. EfficientNetV2 [11] boosts its predecessor and speeds up the training phase using progressive training and exploiting the Fused inverted bottleneck introduced in [14] to reduce the memory overhead in the bottom layers caused by large spatial dimensions. PSLA [24] improves EfficientNet with pretraining, balanced sampling, data augmentation, model aggregation, and attention to obtain a model that achieves new state-of-the-art AudioSet and FSD50K mAPs. [25] introduces the Dynamic Inverted Residual Block composed of the dynamic convolution [26], the dynamic ReLU [27], and Coordinate Attention [28] in place of the Squeeze and Excitation mechanism, to obtain state-of-the-art performance on both AudioSet and FSD50K.

III. Datasets and Data preprocessing

A. Datasets

1) FSD50K: FSD50K (Freesound Dataset 50K) [15] is an open dataset for multi-label sound classification comprising 51197 audio clips between 0.3 and 30 seconds long totaling over 100 hours taken from the Freesound.org project and manually labeled using 200 classes drawn from the AudioSet [19] Ontology. The main limitations of FSD50K are, as in most of the datasets with a high number of samples, Label Noise so incompleteness and/or incorrectness of audio labels, Data Imbalance for which some classes are abundant while others are much less represented, and various biases implicitly introduced to simplify the development of the dataset itself. FSD50K comes already split into training (40966 samples) and test (10231 samples) sets, but for our experiments we further randomly split the training one into a new smaller training set (36796 samples) and a validation set (4170 samples).

2) ESC-50: ESC-50 (Environmental Sound Classification) [16] is an open dataset of 2000 5-second long environmental audio recordings taken from Freesound.org and organized into 50 classes (40 examples per class). The dataset is nowadays considered small and is typically used for assessing model

prediction performance but unlike FSD50K, it is balanced. ESC-50 comes already prearranged into 5 folds for comparable cross-validations.

B. Data preprocessing

We preprocess audios from the two datasets in the same manner. To speed up the training process we use an offline preprocessing pipeline that transforms raw waveforms into normalized audio features in advance before starting the training. Moreover, following [24], we exploit the MixUp technique to improve the generalization abilities over the test set of the models by feeding them, up to some percentage (typically 50%), with mixes of multiple audios and training them with the corresponding mixed labels using an online preprocessing pipeline that mixes random waveforms during the training and finally transforms the resulting into audio features using the offline utilities (that work in parallel on CPU).

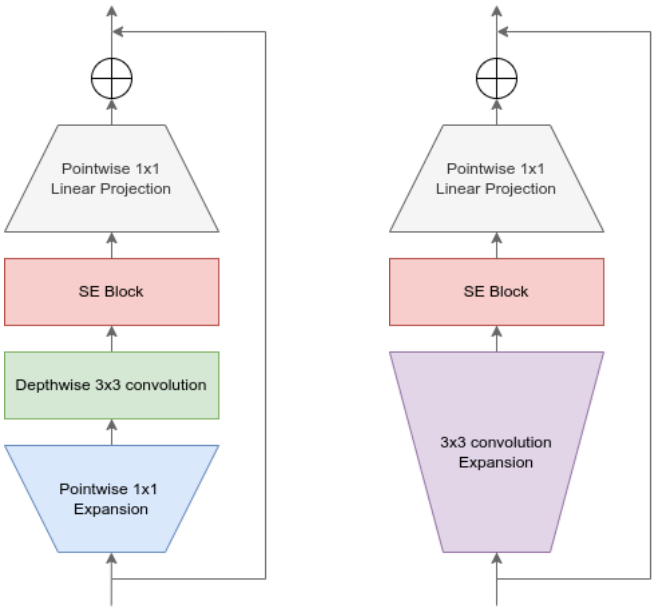
To preprocess the data we follow the literature and resample the raw waveforms to 16kHz with 10s cut, center them, and transform them into 128-dimensional Mel Spectrograms using a 25ms Hanning window that shifts every 10ms. Finally, we standardize the resulting features to achieve more stable training. In the MixUp case, we first average the centered and tiled to the longest randomly chosen waveforms (typically these are 2) and then transform the final waveform, the corresponding labels are averaged as well to obtain the final label. During training, we eventually tile audio features to reach the target length.

IV. Baseline model

Our baseline model is based solely on the MobileNetV3 [9] inverted bottleneck but for the bottom layers that are Fused inverted bottlenecks [14] because, following [11], these blocks are more efficient than the regular ones when spatial dimensions are high, so in the first layers. The standard MobileNetV3 and Fused inverted bottlenecks concepts are shown in Fig. 1 while the baseline model, empirically found, is shown in Fig. 2. In Fig. 2 layers with stride 2 imply a reduction in the spatial dimensions of half of their sizes. All components of the inverted bottlenecks in Fig. 1 are in form of the triplet: convolution layer, batch normalization, SiLU activation function as in [9], but for the Squeeze & Excitation (SE) Block that is composed of an average pooling layer followed by a convolution layer with SiLU activation function and a final convolution layer as described in [18]. The model uses always 2 as the expansion factor for the first component of the inverted bottlenecks but for the first two layers where is, respectively, 32 and 4, so that these blocks internally already operate in the output space. Finally, we set $\frac{1}{4}$ as the SE ratio of all layers of the network, because both empirical results and literature proved it to be a good setting.

V. Splitting and extending the bottleneck

Inspired by the way ResNeXt [12] improved ResNet [13] our approach exploits grouped convolutions to highly reduce



(a) MobileNetV3 inverted bottleneck.

(b) Fused inverted bottleneck [14].

Fig. 1. Blocks of the baseline model. In both (a) and (b) all components are convolution-based layers as described in [9], see [18] for the Squeeze & Excitation (SE) Block.

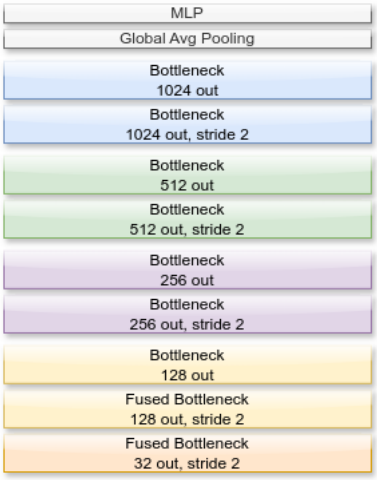


Fig. 2. Architecture of the baseline model.

the complexity of the baseline model and increase its learning proficiency thanks to a wise use of the Squeeze and Excitation (SE) soft-attention mechanism [18]. The method involves replacing, depending on the bottleneck (Fig. 1), either the depthwise separable convolution or the regular convolution and the pointwise convolutions inside the SE Block with corresponding grouped convolutions of the same type, and repeating the last two components — the SE Block and the linear projection — not grouped as final ones: as input passes, if the inverted bottleneck is not fused, each branch expands its set of input channels through a pointwise convolution and then performs a depthwise convolution, while it performs a

single regular convolution if the bottleneck is fused. In both cases, the resulting branch channels are first locally weighted by the SE mechanism and then projected into the branch output space. At this point, all branches collapse into one, and channels are concatenated, these are then globally re-weighed by the SE mechanism in the new space and finally projected into the requested output space. In this way, the soft-attention SE mechanism is applied two times: locally between the channels of each group and globally between the channels of all groups. In Fig. 3 we report our version of the MobileNetV3 inverted bottleneck in both a simplified and compact form, and analogously we do for the Fused inverted bottleneck in Fig. 4. As shown in the figures the final projection of branches is kept linear following the same argument of the original work [8], stating that, non-linearity in the last component of the block destroys information in low-dimensional space. Note also that, when the internal expansion factor of the bottleneck is small, the final projection is not only needed to reach the target output space but also retains most of the bottleneck capacity shrunk by the grouped convolutions. Our approach introduces a new hyperparameter that is analogous to the one called "cardinality" in the ResNeXt work [12] and it represents the number of input channels each branch inside the block has to work with, and that, for simplicity, we call `split` hyperparameter.

VI. Results

In this section, we report the results of the experiments carried out on a Nvidia GTX 1650Ti GPU and an Intel i5-10300H CPU. We trained all of our models using the standard Binary Cross Entropy loss function and Adam optimizer.

A. Complexity, FLOPS and GPU memory

We assess the efficiency of our method, in terms of the number of parameters, FLOPS needed, and GPU memory utilization, when applied to the baseline model for different settings of the `split` hyperparameter by inspecting the results reported in Tab. 1. The approach greatly reduces the complexity of the baseline for all the values of the hyperparameter chosen as well as the FLOPS, thanks to the grouped convolutions in the first part of the bottleneck, at the cost of an increased GPU memory utilization, which we believe is still acceptable in most of the scenarios.

TABLE 1
Comparison of the number of parameters and GPU utilization (FLOPS and memory) between the baseline model and the final model for different choices of the `split` hyperparameter.

	Params	GFLOPS	Total Mem
Baseline	11.4M	3.85	165MB
MergeNet (`split=64`)	9.45M	3.1	220MB
MergeNet (`split=32`)	8.73M	2.56	220MB
MergeNet (`split=16`)	8.36M	2.21	231MB

Given the results in Tab. 1 we decide to choose EfficientNet-B2 [10] and EfficientNetV2-B2 [11] as pure CNNs state-of-the-art models to compare our models with because of the closeness in the number of parameters, as Tab. 2 shows.

TABLE 2
Number of parameters and GPU utilization (FLOPS and memory) of the pure CNNs state-of-the-art models EfficientNet-B2 [10] and EfficientNetV2-B2 [11].

	Params	GFLOPS	Total Mem
EfficientNet-B2	7.91M	1.68	293MB
EfficientNetV2-B2	8.88M	2.98	220MB

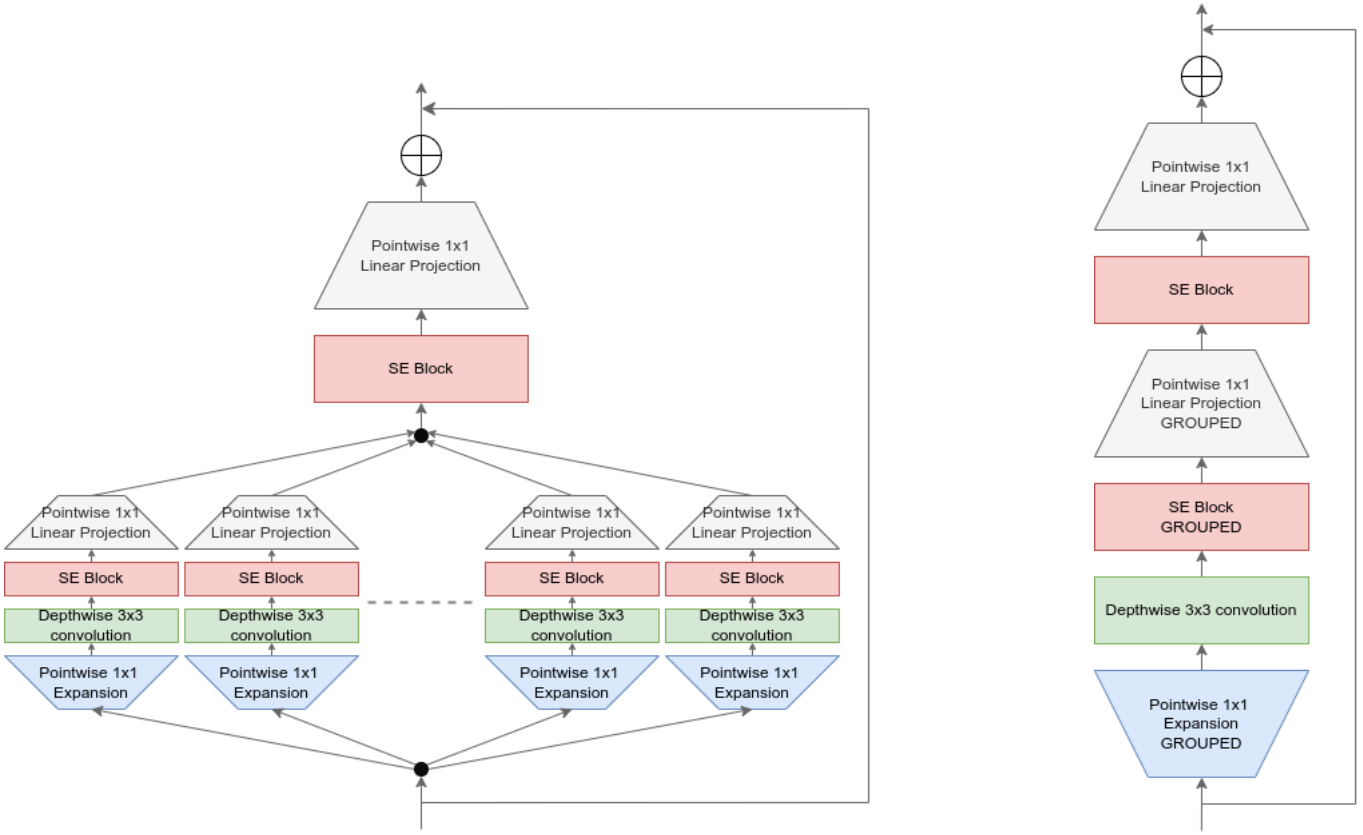
B. FSD50K

The experiments on this dataset all involved training of models with a batch size of 8 for 30 epochs with 0.001 learning rate, and the use of the MixUp technique (`mixup_ratio=0.5`, `number_of_mixed_samples=2`) which proved to be effective with this specific multi-label classification dataset. As an evaluation metric, we use the mAP (mean Average Precision) since is the one used in the original paper ([15]), and in general in the literature, as it summarizes the precision-recall (PR) curve as the classifier decision threshold is varied.

The main experiment involved comparing the prediction performance of the baseline model and the final model for different choices of the `split` hyperparameter. We trained both the baseline and the final models and the two pure CNNs state-of-the-art models EfficientNet-B2 and EfficientNetV2-B2 (see Section VI-A) in the same manner to achieve a fair comparison. The training time was approximately the same for all models and slightly slowed down by the MixUp technique (described in Section III-B), they took $10 - 13$ hours each (≈ 20 minutes per epoch) with our setup. Tab. 3 reports the mAPs on the test set. We observe that our final model with `split=32` achieves the highest mAP on the test set and that, from Section VI-A, it also has the best trade-off between FLOPS needed and GPU memory utilization. From Tab. 3, it is clear how our final models outperform both the baseline and the two state-of-the-art CNNs when they are not pretrained on the large images dataset ImageNet. When the EfficientNet networks are pretrained instead, their mAPs are the highest and it is no wonder, also considering the discussion done about multi-modal pretraining in Section I regarding the work [4].

C. ESC-50

The experiments on ESC-50 were performed with and without an FSD50K pretraining as initialization of the networks weights, as well as, with and without the ImageNet pretraining for the EfficientNet models as in Section VI-B. In the FSD50K pretraining case we just finetuned the models with 5 epochs per fold using a 0.001 learning rate while in the other case,



(a) Simplified concept of our version of the MNv3 inverted bottleneck.

(b) Compact concept of our version of the MNv3 inverted bottleneck.

Fig. 3. Our version of the MNv3 inverted bottleneck. (a) and (b) are functionally equivalent.

TABLE 3
COMPARISON OF THE MAPS ON FSD50K BETWEEN OUR MODELS AND THOSE FROM THE EFFICIENTNET SERIES. THE SECOND COLUMN REPRESENTS THE MAP OF THE MODELS PRETRAINED ON IMAGENET.

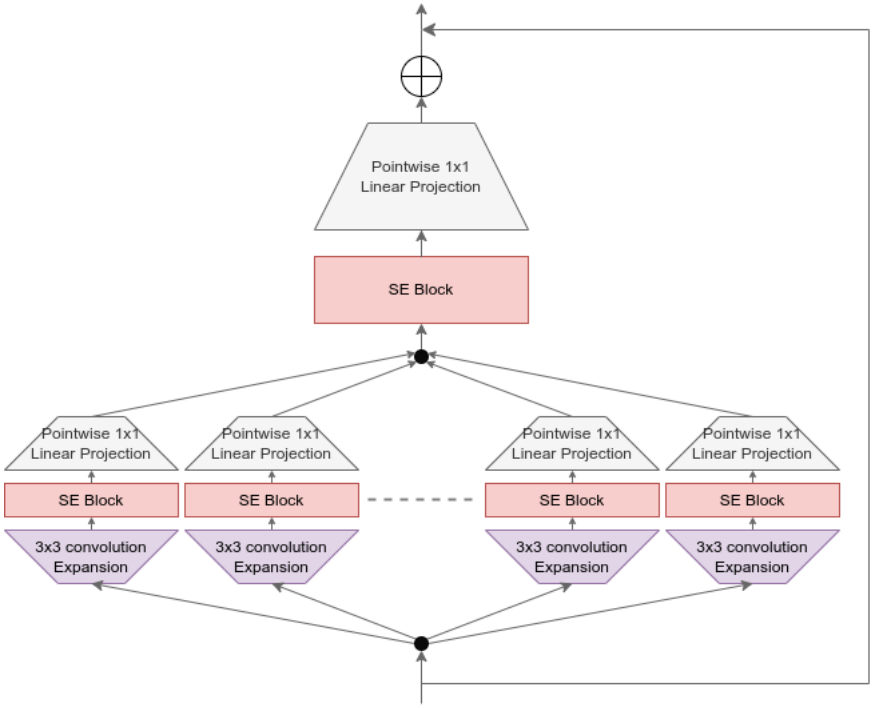
	mAP	mAP w/ ImageNet
Baseline	0.4645	
MergeNet (`split=64`)	0.4790	
MergeNet (`split=32`)	**0.4878**	
MergeNet (`split=16`)	0.4843	
EfficientNet-B2	0.4462	**0.5058**
EfficientNetV2-B2	0.4603	**0.4923**

we trained the models with 10 epochs per fold using a 0.001 learning rate, in both cases we used a batch size of 8. We did not use the MixUp technique with this dataset because it appeared not to return any gain in terms of the final metric, that is the accuracy.

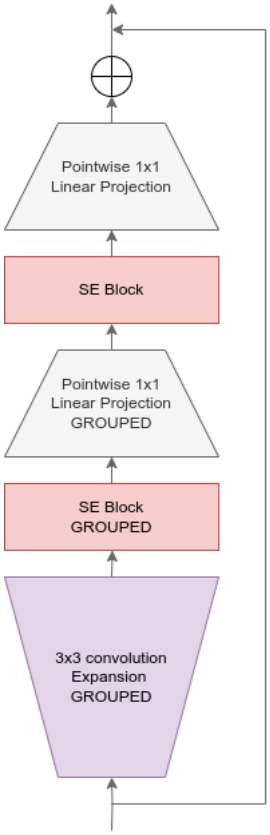
As in Section VI-B, the main experiment involved comparing the prediction performance of the baseline model with that of the final model for different choices of the `split` hyperparameter, and also with those of the pure CNNs state-of-the-art models EfficientNet-B2 and EfficientNetV2-B2 (see Section VI-A), that we trained in the same way as for the other networks to achieve a fair comparison. The training was quick for all models, less than 50 seconds per epoch so no more than 20 minutes for the finetuning and 40 minutes for the regular training. In Tab. 4 we report the results of the experiment. They confirm that the best setting for the hyperparameter `split` for our final model is 32, as found in Section VI-B. Our model achieves the highest accuracy among the models not pretrained on ImageNet and is competitive with those, moreover when pretrained on FSD50K (see Section VI-B) it matches EfficientNet-B2 with ImageNet pretraining accuracy and almost reaches the one of the most performing model, namely EfficientNetV2-B2 with ImageNet pretraining. Fig. 5 compares the performance on this dataset of our best model with those of the current state-of-the-art models, which are mostly transformer-based.

VII. ABLATION STUDIES

In this section, we perform some additional studies to further understand and unveil our approach from different points of view. Note that, in this section, as in Section V and as common in the ResNeXt work [12], we use branch/branches terms to indicate the group/groups of a grouped convolution.



(a) Simplified concept of our version of the Fused inverted bottleneck.



(b) Compact concept of our version of the Fused inverted bottleneck.

Fig. 4. Our version of the Fused inverted bottleneck [14]. (a) and (b) are functionally equivalent.

TABLE 4
COMPARISON OF THE ACCURACIES ON ESC50 BETWEEN OUR MODELS AND THOSE FROM THE EFFICIENTNET SERIES. THE SECOND COLUMN REPRESENTS THE ACCURACY OF THE MODELS PRETRAINED ON FSD50K.

	Acc	Acc w/ FSD50K
Baseline	0.883	0.9665
MergeNet (`split=64`)	0.8675	0.962
MergeNet (`split=32`)	0.8965	0.967
MergeNet (`split=16`)	0.894	0.9665
EfficientNet-B2	0.745	0.957
EfficientNetV2-B2	0.787	0.954
EfficientNet-B2 w/ ImageNet	0.9235	0.967
EfficientNetV2-B2 w/ ImageNet	0.914	0.97

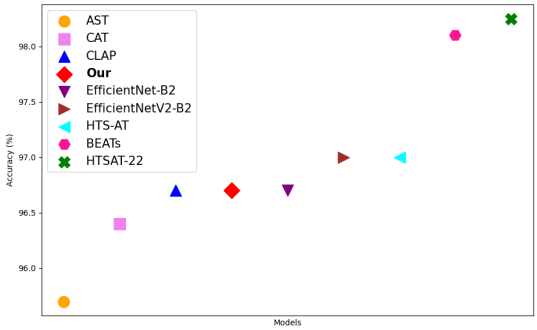


Fig. 5. Comparison of the accuracy of our model on ESC-50 with those of the current state-of-the-art models.

A. Local and global SE

As said and shown in Fig. 3 and Fig. 4, our bottlenecks use the SE mechanism both between the channels of each branch and between the linearly projected channels of all branches. We believe that, apart from the grouped convolutions that reduce complexity, FLOPS, and help in increasing expressiveness of the baseline model, the local and global channels weighing technique made the difference in the results obtained in Section VI, and, in order to prove it, we trained our final model with and without local SE and then, analogously, with and without global SE to assess the differences in classification performance. Tab. 5 reports the results of the experiment. The table shows us what was expected to be, at least, for FSD50K on which the model is less performing when not using one of the two SE while something not

expected on ESC-50, indeed, the model without global SE appears to perform much better, when pretrained on FSD50K, than the full model. This is not that strange when noted that the model without global SE has ≈ 5M neurons (instead of 8M) so that, being small, is able to better fit a small dataset such as ESC-50 but not a bigger one such as FSD50K. We also note that we trained all of our models for the same number of epochs so we do not exclude the possibility that the final model using both SE can achieve 0.975 (or more) accuracy on ESC-50 when trained for more epochs and/or with more advanced training techniques.

TABLE 5
PERFORMANCES OF THE FINAL MODEL WITHOUT LOCAL AND GLOBAL SE MECHANISM. THE FIRST COLUMN REPRESENTS THE MAP ON FSD50K, THE SECOND REPRESENTS THE ACCURACY ON ESC-50, AND FINALLY THE THIRD REPRESENTS THE ACCURACY ON ESC-50 WHEN THE MODEL IS PRETRAINED ON FSD50K.

	mAP	Acc	Acc w/ FSD50K
MergeNet	0.4878	0.8965	0.967
MergeNet w/o local SE.	0.4775	0.8815	0.965
MergeNet w/o global SE.	0.46311	0.892	0.975

B. Importance of channels order

We want to investigate how important the order of channels in input to the different components of the bottleneck is and assess whether the approach we have introduced can satisfy somehow the rotation invariance property w.r.t. the channels. To carry this out we exploit the technique proposed in [29] called Channel Shuffle that with few operations allows us to effectively shuffle the order of the input channels. We use Channel Shuffle before every convolution operation in the bottleneck to enforce the invariance within group kernels and between groups kernels. Tab. 6 reports the results of the corresponding model on both FSD50K and ESC-50. Results show slightly worsened performances on both the datasets indicating some need in having a static ordering of the channels by our approach and, at the same time, the possibility, up to a certain extent, to encourage the rotation invariance w.r.t. to the channels property in our bottleneck, which by its own is not useful but is an expression of the generalization capabilities of our approach.

TABLE 6
PERFORMANCES OF THE FINAL MODEL USING CHANNEL SHUFFLE. THE FIRST COLUMN REPRESENTS THE MAP ON FSD50K, THE SECOND REPRESENTS THE ACCURACY ON ESC-50, AND FINALLY THE THIRD REPRESENTS THE ACCURACY ON ESC-50 WHEN THE MODEL IS PRETRAINED ON FSD50K.

	mAP	Acc	Acc w/ FSD50K
Baseline	0.4645	0.883	0.9665
MergeNet	0.4878	0.8965	0.967
MergeNet + Chan. Shuf.	0.4845	0.8755	0.9665

C. Branch kernel initialization

At the time of implementation, we discovered some discrepancies in the performances of two different implementations of our bottleneck that, at least in theory, had to be functionally equivalent. Further investigation highlighted the importance of the difference between the naive implementation where branches kernels are initialized separately and the final implementation that explicitly exploits grouped convolutions in which kernels are, by default, initialized all together. The differences in performance are reported in Tab. 7 for both datasets. A reasonable explanation for this phenomenon may be that our bottleneck comprises a joining of all output branches channels to perform a global SE pass and final linear projection and for this could need the initialization of kernels to be performed in the original kernel space rather than in that of the single branch one.

TABLE 7
DIFFERENCES IN PERFORMANCE ON BOTH FSD50K AND ESC-50 OF THE FINAL MODEL WHEN USING A SEPARATED BRANCH KERNEL INITIALIZATION. THE FIRST COLUMN REPRESENTS THE MAP ON FSD50K, THE SECOND REPRESENTS THE ACCURACY ON ESC-50, AND FINALLY THE THIRD REPRESENTS THE ACCURACY ON ESC-50 WHEN THE MODEL IS PRETRAINED ON FSD50K.

	mAP	Acc	Acc w/ FSD50K
MergeNet	0.4878	0.8965	0.967
MergeNet w/ sep. init.	0.4849	0.877	0.9605

VIII. CONCLUSIONS

In this work, we proposed a new version of the MobileNetV3 inverted bottleneck that highly reduces the complexity and FLOPS of the original one, and that shows greater expressive power thanks to the use of grouped convolutions and the Squeeze and Excitation soft-attention mechanism used both locally and globally to allow a correct weighing of channels computed in each group and by each group. We also applied the approach to the Fused inverted bottleneck to build a final model, exploiting both the bottlenecks, that is able to achieve competitive performances on both FSD50K and ESC-50 datasets while using fewer FLOPS and neurons than its baseline built with the regular bottlenecks. We compared our results with those of state-of-the-art models belonging to the EfficientNet series to assess the efficiency of our method only challenged by the advantages of pretraining on large datasets. Finally, we performed multiple additional ablation studies to further inspect and understand our method in depth. We believe that our approach can be exploited and investigated more, for example, through the use of advanced baselines built via NAS methods and through the use of pretraining or, even better, multi-modal pretraining.

REFERENCES

[1] S. Srivastava and G. Sharma, "Omnivec: Learning robust representations with cross modal sharing," 2023.

[2] M.-I. Georgescu, E. Fonseca, R. T. Ionescu, M. Lucic, C. Schmid, and A. Arnab, "Audiovisual masked autoencoders," 2024.

[3] S. Chen, Y. Wu, C. Wang, S. Liu, D. Tompkins, Z. Chen, and F. Wei, "Beats: Audio pre-training with acoustic tokenizers," 2022.

[4] K. Palanisamy, D. Singhania, and A. Yao, "Rethinking CNN models for audio classification," *CoRR*, vol. abs/2007.11154, 2020.

[5] F. Schmid, K. Koutini, and G. Widmer, "Efficient large-scale audio tagging via transformer-to-cnn knowledge distillation," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2023.

[6] Y. Gong, S. Khurana, A. Rouditchenko, and J. Glass, "Cmkd: Cnn/transformer-based cross-model knowledge distillation for audio classification," 2022.

[7] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *CoRR*, vol. abs/1704.04861, 2017.

[8] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," *CoRR*, vol. abs/1801.04381, 2018.

[9] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," *CoRR*, vol. abs/1905.02244, 2019.

[10] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019.

[11] M. Tan and Q. V. Le, "Efficientnetv2: Smaller models and faster training," *CoRR*, vol. abs/2104.00298, 2021.

[12] S. Xie, R. B. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," *CoRR*, vol. abs/1611.05431, 2016.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[14] S. Gupta and B. Akin, "Accelerator-aware neural network design using automl," 2020.

[15] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "FSD50K: an open dataset of human-labeled sound events," *CoRR*, vol. abs/2010.00475, 2020.

[16] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *Proceedings of the 23rd ACM International Conference on Multimedia*, MM '15, (New York, NY, USA), pp. 1015–1018, Association for Computing Machinery, 2015.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *CoRR*, vol. abs/1409.0575, 2014.

[18] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *CoRR*, vol. abs/1709.01507, 2017.

[19] 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 *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 776–780, 2017.

[20] Y. Gong, Y. Chung, and J. R. Glass, "AST: audio spectrogram transformer," *CoRR*, vol. abs/2104.01778, 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," *CoRR*, vol. abs/2010.11929, 2020.

[22] Y. Gong, C. J. Lai, Y. Chung, and J. R. Glass, "SSAST: self-supervised audio spectrogram transformer," *CoRR*, vol. abs/2110.09784, 2021.

[23] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely connected convolutional networks," *CoRR*, vol. abs/1608.06993, 2016.

[24] Y. Gong, Y. Chung, and J. R. Glass, "PSLA: improving audio event classification with pretraining, sampling, labeling, and aggregation," *CoRR*, vol. abs/2102.01243, 2021.

[25] F. Schmid, K. Koutini, and G. Widmer, "Dynamic convolutional neural networks as efficient pre-trained audio models," 2023.

[26] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic convolution: Attention over convolution kernels," *CoRR*, vol. abs/1912.03458, 2019.

[27] Y. Chen, X. Dai, M. Liu, D. Chen, L. Yuan, and Z. Liu, "Dynamic relu," *CoRR*, vol. abs/2003.10027, 2020.

[28] Q. Hou, D. Zhou, and J. Feng, "Coordinate attention for efficient mobile network design," *CoRR*, vol. abs/2103.02907, 2021.

[29] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," *CoRR*, vol. abs/1707.01083, 2017.

On what I have learned from this project

I learned to understand the importance of the complexity, FLOPS, and GPU memory utilization of both a module and a model when budgets are limited. I have better understood the convolution operation and its possible variations when dealing with a real-world problem where hardware matters. I have understood how important is research, and exploiting and combining others' results to let the field move on.

On the problems I have encountered in this project

I have not encountered big problems. Of course, the hardware setup was poor and working alone limiting but it has been challenging and evolutionary.