# Optimizing PhiNet architectures for the detection of urban sounds on low-end devices

Alessio Brutti
*Digis Center*
*Fondazione Bruno Kessler*
Trento, Italy
brutti@fbk.eu

Francesco Paissan
*Digis Center*
*Fondazione Bruno Kessler*
Trento, Italy
fpaissan@fbk.eu

Alberto Ancilotto
*Digis Center*
*Fondazione Bruno Kessler*
Trento, Italy
aancilotto@fbk.eu

Elisabetta Farella
*Digis Center*
*Fondazione Bruno Kessler*
Trento, Italy
efarella@fbk.eu

***Abstract*—Sound Event Detection (SED) pipelines identify and classify relevant events in audio streams. With typical applications in the smart city domain (e.g., crowd counting, alarm triggering), SED is an asset for municipalities and law enforcement agencies. Given the large size of the areas to be monitored and the amount of data generated by the IoT sensors, large models running on centralised servers are not suitable for real-time applications. Conversely, performing SED directly on pervasive embedded devices is very attractive in terms of energy consumption, bandwidth requirements and privacy preservation. In a previous manuscript, we proposed scalable backbones from the PhiNets architectures' family for real-time sound event detection on microcontrollers. In this paper, we extend our analysis investigating how PhiNets' scaling parameters affect the model performance in the SED task while searching for the best configuration given the computational constraints. Experimental analysis on UrbanSound8K shows that while only the total number of parameters matters when training the model from scratch (i.e., it is independent of the scaling parameter configuration), knowledge distillation is more effective with specific scaling configurations.**

***Index Terms*—Sound event detection, Neural Networks, PhiNets, tinyML**

## I. Introduction

Sound Event Detection (SED) is an emerging task with many applications in fields like industries and intelligent cities [1], where multimedia analytics gained significant interest in the recent past [2], [3]. SED can benefit from the availability of pervasive embedded devices capable of continuously monitoring the environment looking for relevant events [1]. Driven by the release of novel datasets and challenges [4]–[7], recent advancements in the field have considerably improved the effectiveness and accuracy of SED solutions. However, this has been achieved using highly demanding models in terms of memory footprint and computational complexity [8]–[13]. Consequently, these systems are not suitable for applications requiring pervasive low-power, low-cost sensors. Nonetheless, it has been shown how strategies such as knowledge distillation (KD) [14], network pruning [15] or weight quantization [16], [17] can considerably reduce the size of the models, making them suitable to run on microcontroller units (MCU) [18]. Unfortunately, these techniques are typically tailored to the specific device and require an expensive process to adapt the original neural network to different processing units. Therefore, efforts have been recently focused on developing architectures specifically designed to operate on low-end devices [19].

Following this line of research, in a previous work [20] we applied *PhiNets* [21] to the SED task either using spectrograms or raw waveforms. The proposed model achieved state-of-the-art performance on the UrbanSound8k dataset [4] for spectrogram classification while using an extremely low number of parameters. The focus of this previous paper was on minimising as much as possible the memory footprint of the models, in order to make it fit on MCUs, while limiting the performance deterioration with respect to the state-of-the-art. Conversely, in this paper, we provide an experimental analysis on how the PhiNet's width-scaling parameters (namely the width multiplier $\alpha$ and the base expansion factor $t_0$ from the original paper [21]) impact the final classification performance by defining models of different sizes and architectures. In particular, we observed that different configurations of the scaling parameters leading to the same amount of model parameters give very similar performance. On the other hand, applying KD from a larger teacher model boosts the performance but only when large $t_0$ values are employed.

The paper is organised as follows. Section II describes the PhiNet backbones and their scaling parameters. Section III provides details about the experimental analysis whose results are reported and discussed in Section IV. Finally, Section V concludes the paper with final remarks.

## II. Scalable backbones: PhiNets

This work employs the PhiNets networks [21]: a family of modular scalable backbones that can be easily tuned using few hyperparamters to match the memory and computational resources available on different embedded platforms. The main convolutional block used in the architecture is a modified version of the inverted residual block used in MobileNetV2 [22] and MobileNetV3 [23] architectures. This block is composed of a sequence of three operations, namely: a pointwise *expansion* convolution, a depthwise convolution, a squeeze-and-excitation block [24]. These three operations are followed by a second pointwise *projection* convolution. The structure

This work was partially funded by the EU H2020 project MARVEL (project number: 957337).

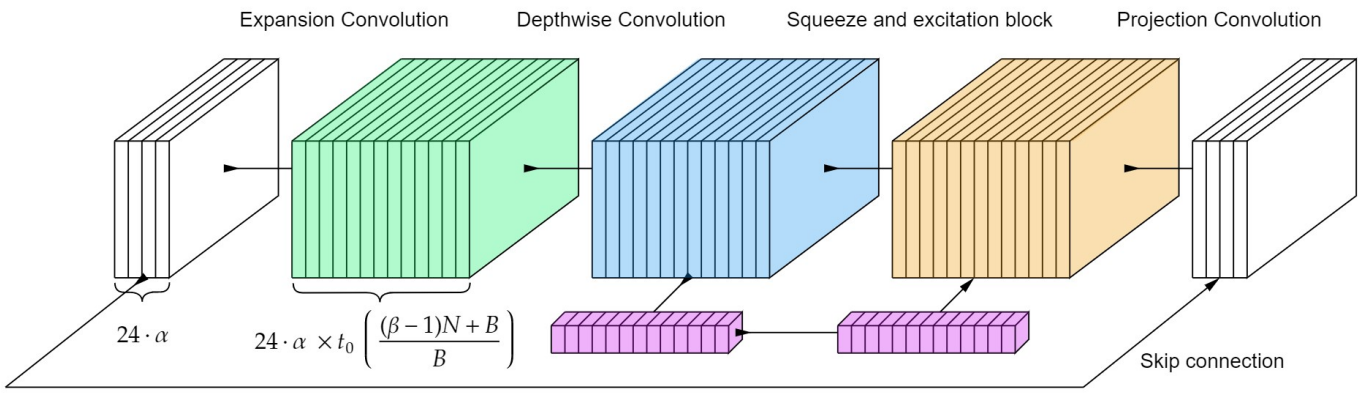


Fig. 1. An overview of the PhiNets convolutional block structure. First, the number of channels is increased with a pointwise convolution, followed by a depthwise convolution (green) and SE block (blue). Finally, a second pointwise convolution (yellow) connects to the low dimensionality bottleneck block (purple). $B$ is the number of blocks in the network, and $N$ is the current block index. $\alpha$, $\beta$ and $t_0$ are the block hyperparameters.

of the basic PhiNet convolutional block is shown in Fig. 1. The final model is obtained stacking $B$ of these blocks (we use $B = 5$ for all the experiments in this work). Three hyperparameters can be used to modify the configuration of the convolutional blocks:

- **Width multiplier** $\alpha$, which linearly adjusts the filter count of all convolutions in the network. As a result, it scales the operation count of the whole model. The number of operations in the network depends quadratically on this parameter, as shown in Fig. 2; This quadratic dependence can be verified experimentally with most deep learning platforms. Moreover, it can be trivially derived considering that the operation count for one convolutional block is:

$$\text{MMAC}(c_i, c_o, \alpha, n) = c_i c_o \alpha^2 n^2 \tag{1}$$

  where $c_i$, $c_o$, $\alpha$ and $n$ are the input channels, output channels, width multiplier and input size respectively.

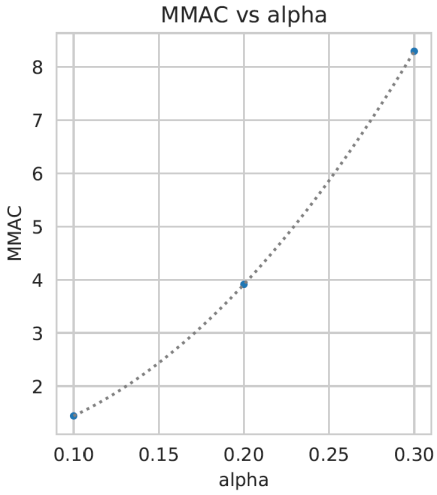


Fig. 2. Effects of varying the hyperparameter $\alpha$ on network operations , expressed in Multiply and Accumulate, (MACC).

- **Base expansion factor** $t_0$, which affects the filter count in the expansion and depthwise convolutions inner blocks. This parameters can be used to optimise the RAM required by the network, which can be approximated as $R \approx C \cdot t_0$ with $C$ denoting the RAM needed for the network with $t_0 = 1$. The effects of this parameter on the network working memory is shown in Fig. 3;

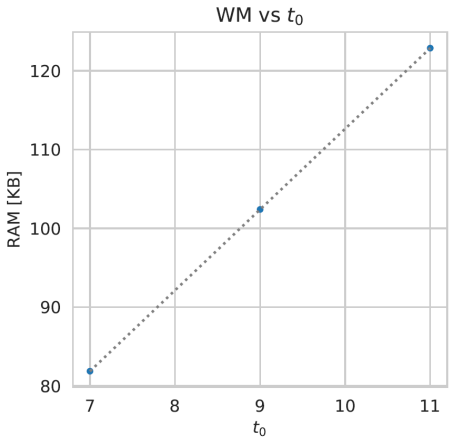


Fig. 3. Effects of varying the hyperparameter $t_0$ on the working memory (WM) or RAM required to store the intermediate network tensors.

- **Shape parameter** $\beta$, that defines the filter count of the later blocks in the networks. These blocks are those the ones requiring the largest number of parameters, which can be approximated as $\#Params \approx C \cdot \frac{1}{2}(1+\beta)$, where $C$ is number of parameters of the network with $\beta = 1$. The effects of this parameter on network parameters are shown in Fig 4.

The computational cost and memory footprint of the model can be easily adjusted to fit the constraints of the processing unit by varying these three hyper-parameters. Note that different configurations of the hyper-parameters may lead to highly similar parameter counts but rather different architectures. In this work, we want to investigate the effectiveness of

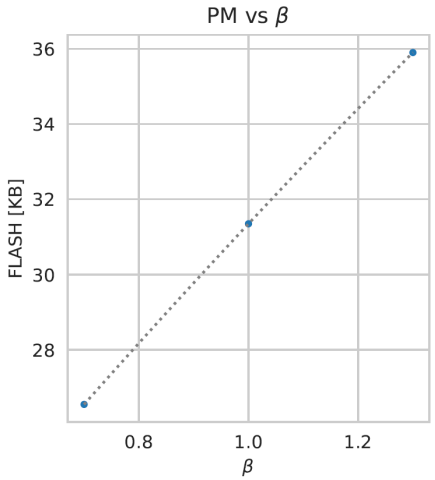


Fig. 4. Effects of varying the hyperparameter $\beta$ on the parameter memory (PM) or FLASH required to store the network weights.

these different configurations. The sequence of blocks is then followed by a fully connected classification layer with softmax activation in order to obtain a 10 class classification output.

## III. Experimental Analysis

This work carries out an empirical study to highlight the effects of two width scaling parameters ($\alpha$ and $t_0$) on sound event detection performance. $\beta$ will be kept at the default value ($\beta = 1$), as all networks tested require so few parameters that even the smallest MCUs can store them with a considerable margin. We vary $\alpha$ considering [0.20, 0.35, 0.50] possible values and $t_0$ in [2, 4, 6]. Note that in this way, we cover different architectures with a very similar number of parameters. Given the small sizes of the resulting PhiNet models, besides training them from scratch, we also investigate the use of KD from a larger plain-conv2d model, using both soft and one-hot labels.

### A. Dataset

We perform our analysis on the UrbanSound8K dataset [4], a collection of 8732 samples of 4 second long typical urban sound events, equally distributed among 10 different classes (air conditioner, car horn, children playing, dog bark, drilling, engine idling, gunshot, jackhammer, siren, and street music). We re-sampled each event at 16 kHz. Moreover, we augmented the dataset with pitch shifting with tone steps -2, -1, 1, 2, time-stretching with factors 0.81 and 1.07, and additional Gaussian noise. The task is single-label classification, and the performance is hence evaluated in terms of classification accuracy. We used the standard 10-fold benchmarking procedure for this dataset.

### B. Implementation Details

The model input is 2D and consists of 40 mel-spectrum features computed on the 4 s segments using a 128 ms sample window with a hop-length 42 ms. Overall 120 frames are computed for each sound event. We trained all models for 200 epochs, with a $1 \times 10^{-3}$ learning rate, $1 \times 10^{-2}$ weight decay and 0.07 dropout rate in the convolutional blocks.

We used a plain conv-2d model consisting of 4 conv-2d layers with 16, 32, 64, 64 filters each for the KD-based training. After the convolutional layers, we used a softmax classifier. The loss was a combination of the cross-entropy computed on the teacher's soft labels and the hard labels given by the ground-truth, with a ratio of 2/3-1/3. The temperature parameter was set to 2. Both hyper-parameters were empirically optimised.

## IV. Results

Table I reports the sound event detection accuracy obtained training the models from scratch, as well as using knowledge distillation, considering different configurations. The table also reports the parameter count for each configuration.

TABLE I
Accuracy on UrbanSound8K varying the $\alpha$ and $t0$ scaling parameters, with and wihtout KD. The table reports also the model parameter count.

| $\alpha$ | $t0$ | Acc | Acc-KD | # Parameters |
|---|---|---|---|---|
| 0.20 | 2 | 64.87 | 49.68 | 4,779 |
| 0.35 | 2 | 64.64 | 64.82 | 12,479 |
| 0.50 | 2 | 63.90 | 67.61 | 24,507 |
| 0.20 | 4 | 65.85 | 59.58 | 8,893 |
| 0.35 | 4 | 71.80 | 67.14 | 23,797 |
| 0.50 | 4 | 72.25 | 70.15 | 47,349 |
| 0.20 | 6 | 66.05 | 70.95 | 13,007 |
| 0.35 | 6 | 68.02 | 71.05 | 35,115 |
| 0.50 | 6 | 70.39 | 71.20 | 70,191 |

While the performance of the models trained from scratch decays rather linearly with the number of parameters (as shown in Fig. 5), the specific configuration of the hyper-parameters $\alpha$ and $t_0$ does not seem to have a direct and evident impact on the performance (see Fig 7, 6). Overall, this was expected as the PhiNet architectures are designed to scale efficiently in the MCU range without significantly compromising the network's performance. However, it is worth noting that, in some cases, using larger values of $t_0$ is preferable with respect to $\alpha$ given a target parameter count (compare for example the two models (0.5;2) and (0.35;34)). This is true also considering that both $\alpha$ and $t_0$ have a quadratic dependency on inference time. This could be related to the fact that larger convolutional blocks can better represent the information, easing the learning task. Finally, note that the best accuracy (72.25%) is achieved using a medium-size architecture (47K parameters obtained with $\alpha = 0.5$ and $t_0 = 4$). PhiNets are actually designed to be efficient in the MCU range. Therefore, they tend to overfit easily when the model size increases. This issue is further accentuated by the relatively small size of the dataset used in our experiments.

Conversely, models trained via knowledge distillation do not show the same linear performance decay that, instead, drastically decreases for small architectures. Nevertheless, the experimental analysis confirms that KD improves the performance of some configurations. However, this occurs only when the expansion factor is sufficiently high ($t_0 = 6$ in

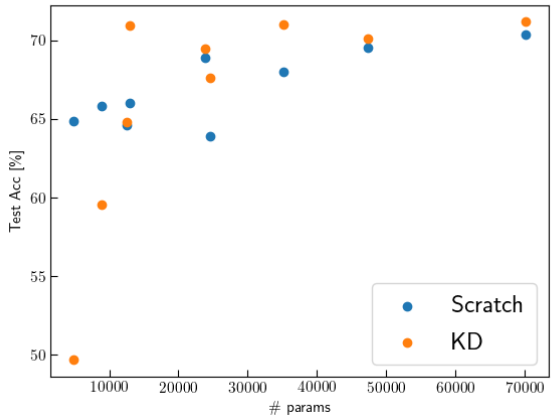


Fig. 5. Event detection accuracy as a function of the overall network parameter count.

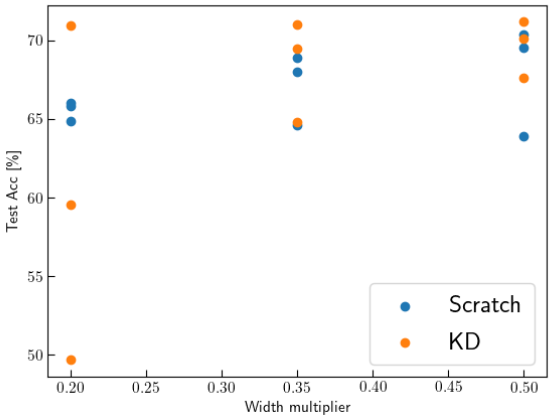


Fig. 7. Testing accuracy as a function of $\alpha$. Again, we observe that higher values for the width multiplier is preferable when training using knowledge distillation.

our experiments). In all the other cases, using KD leads to performance deterioration, which in some cases are extremely evident. Our hypothesis is that architectures with high values of $t_0$ tend to resemble the conv-2d nature of the teacher, thus helping the convergence of the model.

Overall, the scaling principles of PhiNets guarantee a competitive classification accuracy without requiring KD.

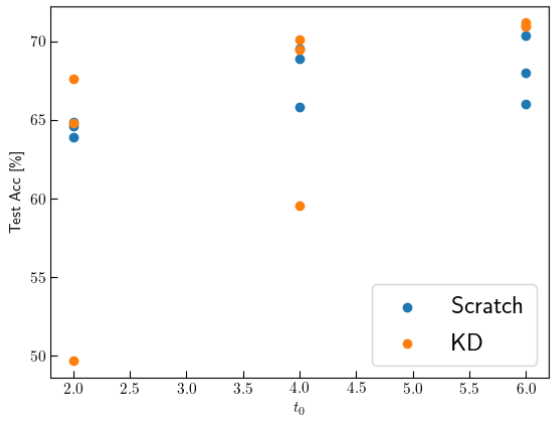


Fig. 6. Classification accuracy as a function of $t_0$. Note how a higher value for the base expansion factor favors networks trained using knowledge distillation.

To complete our analysis, in Table IV we compare the best performance achieved with PhiNets with the state-of-the-art models AudioCLIP [8] and with the plain conv-2d model used as teacher. Note that the very high performance of AudioCLIP is achieved with 60M parameters, which are definitely not suitable for low-end devices. In addition AudioCLIP has been trained on a much larger amount of data, while our models are trained direclty on UrbanSound8K. Nevertheless, the proposed PhiNet architecture can reach a 72.2% accuracy with less the 50K parameters. It is interesting to observe that the plain conv-2d teacher considerably outperforms the PhiNet model with a similar parameter count (see the last row of Table I). This is mainly due to the fact that the teacher network composed of 2D convolutions requires largely more operations to run with respect to the largest PhiNet tested (90M vs 10M).

TABLE II
MODEL PERFORMANCE WITH RESPECT TO STATE-OF-THE-ART PLATFORM.

| Model name | Test acc [%] | Parameter count |
|---|---|---|
| AudioCLIP [8] | 90.01 | 60M |
| Teacher (Conv-2d) | 81.15 | 66K |
| PhiNets | 72.25 | 47K |

## V. CONCLUSIONS

In this paper, we investigated the impact of two width-scaling parameters of PhiNets towards identifying their effects on the performance of urban sound detection to simplify the model design given the available memory and computational resources. Experiments on UrbanSound8K show that while $\alpha$ and $t_0$ are interchangeable when the model is trained from scratch, and only the number of parameters matters, large values of $t_0$ are preferable if KD from a pre-trained teacher model can be applied.

In future work, we aim at applying the same KD approach on a video task to validate the results against a different sensing source. Moreover, we plan to compare this approach with adaptive pruning strategies for network compression.

## REFERENCES

[1] D. Bajovic *et al.*, "Marvel: Multimodal extreme scale data analytics for smart cities environments," in *2021 International Balkan Conference on Communications and Networking (BalkanCom)*, 2021, pp. 143–147.

[2] F. Paissan, G. Cerutti, M. Gottardi, and E. Farella, "People/car classification using an ultra-low-power smart vision sensor," in *2019 IEEE 8th International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2019, pp. 91–96.

[3] F. Paissan, M. Gottardi, and E. Farella, "Enabling energy efficient machine learning on a ultra-low-power vision sensor for iot," *arXiv preprint arXiv:2102.01340*, 2021.

[4] J. Salamon, C. Jacoby, and J. P. Bello, "A dataset and taxonomy for urban sound research," in *22nd ACM International Conference on Multimedia (ACM-MM'14)*, Orlando, FL, USA, Nov. 2014, pp. 1041–1044.

[5] 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 *ICASSP*, March 2017, pp. 776–780.

[6] K. J. Piczak, "Esc: Dataset for environmental sound classification," in *ACM international conference on Multimedia*, 2015, pp. 1015–1018.

[7] A. Mesaros, T. Heittola, A. Diment, B. Elizalde, A. Shah, E. Vincent, B. Raj, and T. Virtanen, "DCASE 2017 challenge setup: Tasks, datasets and baseline system," in *DCASE*, 2017.

[8] A. Guzhov, F. Raue, J. Hees, and A. Dengel, "Audioclip: Extending clip to image, text and audio," *arXiv preprint arXiv:2106.13043*, 2021.

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

[10] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "Cnn architectures for large-scale audio classification," in *2017 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2017, pp. 131–135.

[11] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[12] Y. Tokozume and T. Harada, "Learning environmental sounds with end-to-end convolutional neural network," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2721–2725, 2017.

[13] D. W. Romero, E. J. Bekkers, J. M. Tomczak, and M. Hoogendoorn, "Wavelet networks: Scale equivariant learning from raw waveforms," *arXiv preprint arXiv:2006.05259*, 2020.

[14] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, 2015.

[15] L. Valerio, F. M. Nardini, A. Passarella, and R. Perego, "Dynamic hard pruning of neural networks at the edge of the internet," *arXiv preprint arXiv:2011.08545*, 2020.

[16] G. Cerutti, R. Andri, L. Cavigelli, E. Farella, M. Magno, and L. Benini, "Sound event detection with binary neural networks on tightly power-constrained iot devices," in *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, 2020, pp. 19–24.

[17] J. Yang, X. Shen, J. Xing, X. Tian, H. Li, B. Deng, J. Huang, and X.-s. Hua, "Quantization networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7308–7316.

[18] G. Cerutti, R. Prasad, A. Brutti, and E. Farella, "Compact recurrent neural networks for acoustic event detection on low-energy low-complexity platforms," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 4, pp. 654–664, 2020.

[19] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions," *arXiv preprint arXiv:2005.14623*, 2020.

[20] F. Paissan and E. F. Alberto Ancilotto, Alessio Brutti, "Scalable neural architectures for end-to-end environmental sound classification," *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022.

[21] F. Paissan, A. Ancilotto, and E. Farella, "PhiNets: a scalable backbone for low-power AI at the edge," *arXiv preprint arXiv:2110.00337*, 2021.

[22] 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.

[23] 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.

[24] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *CoRR*, vol. abs/1709.01507, 2017. [Online]. Available: http://arxiv.org/abs/1709.01507