FlyNet: Max it, Excite it, Quantize it

Luis Guerra[1]
luis.guerrafernandez@monash.edu
Tom Drummond[2]

[1] Monash University
Melbourne, Australia
[2] The University of Melbourne
Melbourne, Australia

Abstract

We present a new efficient Convolutional Neural Network (CNN) architecture targeted for tiny machine learning (TinyML) on vision tasks. Through a series of architectural improvements to state-of-the-art mobile networks, we are able to significantly reduce the number of parameters while maintaining a reasonable number of multiply-accumulate operations. We switch the load from expensive pointwise convolutions into lighter multihead-depthwise convolutions with non-linear Max-out aggregation. By incorporating our contributions to a MobileNetV3 backbone, we achieved comparable accuracy with up to 0.5x reduction in parameters in the ImageNet dataset, and achieved 61% top-1 accuracy, matching MicroNet-M2, ShufflenetV2-0.5x and EfficientNet-B, with 1MB. We additionally reported results in the tasks of object detection in the COCO dataset. Finally we performed ablation studies to demonstrate the effectiveness of our improvements. Code will be made available online. Code available at: http://github.com/luis-guerraf/flynet

1 Introduction

Machine learning targeted for highly constrained devices is a topic of growing interest as industry demands for real-time, lightweight solutions deployable in IoT devices. Recent approaches to tackle this problem involve reducing the amount of operations [37, 42, 56]; however, reducing the number of parameters has, to some extent, been overlooked. Nevertheless, as described by Xu *et al.* [66], transmitting model parameters across different hierarchies of memories in an embedded device can account for up to 80% of the power consumption of an inference routine. For example, a commercial microcontroller might not fit a state-of-the-art efficient network in on-chip memory [1].

Similarly a network designed for constant operation on an always-on device, and a network that will be loaded on demand by a cell-phone app to perform a single inference cycle, will have different energy profiles. The first one might be better suited for a low FLOPs model, whereas the second one for a low parameters model.

Currently there is an ongoing paradigm switch in field of deep learning migrating from once ubiquitous CNNs to Transformer-based [60] neural architectures [13, 17, 31] and MLP-only variants [57, 58]. Nevertheless, convolutional backbones are still leading the board in parameter efficiency [37, 38, 42]. In an effort to design light networks several directions have been proposed including low rank matrix decompositions [12, 70], pruning [22, 74] and sparsification [36, 41], quantization [9, 47], architecture search [40, 75, 76], weight sharing [20], efficient building blocks design [8, 25] and linear self-attention [32, 44]. Notable works have relied on depthwise separarable convolutions [51], such as the Xception [8] and MobileNets series of architectures [24, 49]. Architectures such as ResNeXt [65] and ShuffleNet [42, 71] rely on group convolutions [35] while others on attention mechanisms, such as Squeeze-and-Excitation networks [27, 63], and residual connections [21] among others.

In this work we propose four simple but highly effective architectural modifications to mobile networks that can be used as building blocks in further architectures exploration. Concretely, we switched the load from the expensive pointwise convolutions to the much lighter *multihead-depthwise convolutions* which generalizes depthwise convolutions. By aditionally using a max(·) activation function as feature aggregation, our method is equivalent to a Maxout network [15] with depthwise convolutions. Additionally, we added efficient mean and variance aware channelwise attention. We leveraged dense residual connections, in contrast to previous MobileNet versions [24, 49] by simply cropping and padding where necessary, and finally we regularized our networks by injecting quantization noise to improve generalization.

These contributions grant us significant accuracy gains over already well established mobile architectures pushing the barrier towards extremely low power and highly efficient CNNs for embedded devices. We achieved comparable accuracy to MobileNetV3 [24] with up to 0.5x reduction in parameters in the ImageNet dataset, and achieved 61% top-1 accuracy, matching MicroNet-M2 [37], ShuffleNetV2-0.5x [42] and EfficientNet-B [56], with 1MB. We verified the efficacy of our method in the task of COCO object detection, and performed extensive ablation studies to demonstrate the effectiveness of each of our contributions.

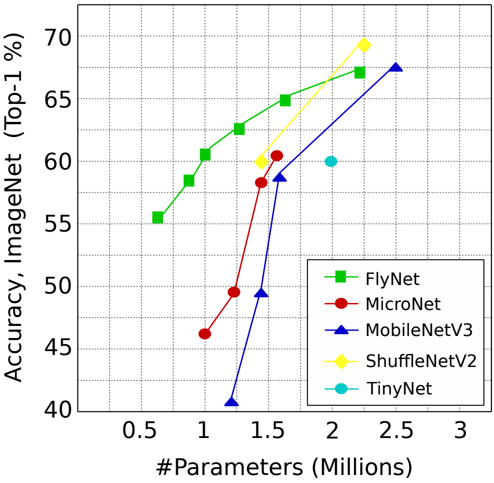


Figure 1: Comparison of FlyNet-h3 (3 MDW heads) architecture with different width multipliers against state-of-the-art models. FlyNet performs favorably in terms of parameter count against both popular and recently published works. Particularly, our method excels as the model shrinks, towards the extremely low parameter regime.

2 Related Work

Efficient Building Blocks. With the intention of reducing computation and storage in CNNs, several works in literature have proposed different neural building blocks. An initial attempt was the depthwise separable convolutions presented in the Inception series [8, 55] and MobileNets [24, 25, 49], followed by SqueezeNet's fire module [29]. Group convolutions were utilized in ResNeXt [65] along with channel shuffling in ShuffleNet [42, 71]. The linear bottleneck was introduced in MobileNetV2 [49]. Finally, residual connections [21, 53], although initially conceptualized to deal with vanishing gradients, have been utilized as means to obtain increases in performance with little overhead as in the sandglass block [72]. In this work a new block denoted *multihead-depthwise convolutions* is introduced, which we use extensively across FlyNet, along with dense lightweight residuals. Recently this block, also denoted generalized depthwise-separable convolution, along with concatenation was used for adversarially robust and efficient networks [11].

Dynamic Convolution and Mixture of Experts. Our method holds similarity to mixture of experts based models [16, 45, 50] where the outputs of different convolutional filters are fused by a gating or weighting mechanism with the main difference being that our fusing function is a differentiable non-linearity. In contrast, the more recently proposed dynamic convolutions [4, 67] adaptively adjust the convolutional filters prior to performing the convolution operation. Thus, the number of operations remains almost unchanged; however, the network grows considerably in size which is non-trivial given that typically memory accesses account for a considerable amount of power consumption [23]. Our approach differs from dynamic convolution in the sense that the filter responses are computed and aggregated via a Max-out activation function with squeeze-and-excitation [27] scalings. We only implement this strategy in depthwise convolutions, offloading the pointwise convolutions in MobileNet.

Efficient Attention. Squeeze-and-Excitation networks (SENet), introduced in [27], pioneered a whole set of attention based inexpensive architectural improvements including [6, 26, 46, 54, 63, 64]. Recently [61] proposed replacing the multi-layer perceptron (MLP) in SENet for a single convolution. In this work we endow this lightweight channel attention with variance awareness, translating in accuracy gains at little extra overhead. Second-order statistics with channel-wise attention had been considered before in [14]; however, considerable complexity is added for similar gains to our approach as the authors consider the entire channels covariance matrix.

Improving Generalization by Injecting Noise. Over-fitting to the training data is an issue that has been extensively explored from different angles [10, 33, 34, 48, 69]. Dropout [52] has been modelled as a Gausssian noise injection process during the learning process [62]. Batch-norm [30], originally believed to reduce the internal covariate shift was recently shown to be successful in part due to the same noise-injecting principle in the form of normal additive noise and scaled inverse Chi multiplicative noise which depend on the batch size [68]. Here, we present noise regularization based on features and parameters quantization. To the best of our knowledge, this is the first time quantization has been approached from a regularization perspective.

3 FlyNet

For this work, we relied on MobileNetV3 [24] as base architecture for implementing our contributions described in the following subsection. Then we enlist our contributions and elaborate on each of them.

3.1 MobileNetV3 Backbone and Motivation

Popular compressed architectures such as MobilNetV3 and EfficientNet [56] based themselves on the MobilenetV2 [49] architecture, consist of a stack of depthwise separable convolutions and linear bottlenecks. Depthwise separable convolutions are comprised by depthwise and pointwise convolutions which is a form of factorization of a standard convolution. Depthwise convolutions, implemented as *3x3 groups=1* convolutions, have the function of finding spatial patterns but have no cross-channel communication. Pointwise convolutions, implemented by *1x1 groups=m* convolutions, have the function of mixing information across feature channels. The linear bottlenecks are implemented as two sequential pointwise convolutions that compress and expand the features with no activation function. Finally, inverted residuals connect the compressed representations in the linear bottlenecks allowing gradients to flow better throughout the network.

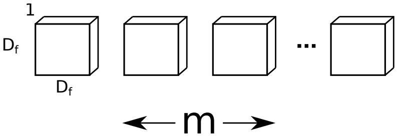


Figure 2: Standard depthwise features

Borrowing notation from [49], a convolutional layer has the following parameters: D_k^2, D_f^2, m and n that denote the dimension of a convolutional kernel, dimension of the input feature map, number of input channels and number of output channels, respectively. Depthwise convolutions have a computational cost of $D_k^2 \cdot m \cdot D_f^2$, and require $D_k^2 \cdot m$ weights. Pointwise convolutions have a computational cost of $m \cdot n \cdot D_f^2$, and require $m \cdot n$ weights. It is easy to see that pointwise convolutions are far more expensive than depthwise ones, specially as m and n grow with the depth of the network. Based on this observations we propose *multihead-depthwise convolutions* to offload pointwise convolutions while attempting to maintain the representation capacity of the network.

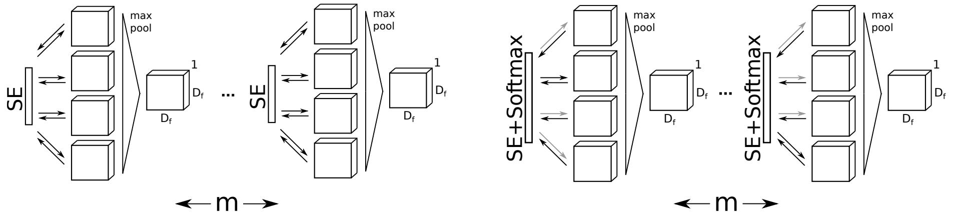
3.2 Architectural Contributions

Multihead-Depthwise Convolutions with Max-out Activation.

Multihead-depthwise (MDW) convolutions are a generalization of depthwise convolutions that leverage the over-looked output channels dimension (output depth). Whereas traditionally in a depthwise convolution each input channel gets convolved with only a single filter, in a MDW convolution it gets convolved with $h \geq 1$ filters. The resulting $m \cdot h$ output feature maps can subsequently be merged in several ways. A naive approach would involve using a pointwise convolution to shrink the dimension back down to the original dimension (denoted input depth in [25]). An appealing second alternative is to linearly combine each set of h heads produced by each input channel using a weighted combination followed by an activation function (*e.g.* relu$(\cdot)$). This approach incurs in fewer parameters and computation. However, in this work we propose to use the *Max-out* activation function proposed in [15], which is a parameter-free, non-linear aggregation, involves no Multiply-Add operations (MAdds) and achieves higher accuracy than a learned affine transformation. Refer to Table 1 for ablations on types of reductions and section 4.3 on number of heads.

Formally, given an input features tensor $A \in \mathbb{R}^{m \times f_1 \times f_2}$, MDW convolution convolves it with a h-heads kernel $W \in \mathbb{R}^{m \cdot h \times k_1 \times k_2}$ to produce pre-activations $F_{pre} \in \mathbb{R}^{m \times h \times f_1 \times f_2}$ and subsequently reduce them with $\max_{\mathrm{dim}=1}(F)$ to produce post-activations $F_{post} \in \mathbb{R}^{m \times f_1 \times f_2}$. This is illustrated in Figure 3. As in [43], in order to implement a *Funnel Max-out* activation, we additionally tested adding the input feature maps A as input to the Max-out activation: $\max_{\mathrm{dim}=1}([A, F_{\mathrm{pre}}]_{\mathrm{dim}=1})$ with no success. $\max_{\mathrm{dim=j}}(\cdot)$ and $[\cdot]_{dim=j}$ denote max pooling and concatenation across dimension j with dimensions indexing starting at zero. Refer to section 3.2 for the results.

The computational cost of MDW convolution is simply h times that of a regular depth-



(a) Multihead-depthwise (MDW) features with Max-out aggregation

(b) MDW with softmax excitation

Figure 3: (a)Act as feature map **mixer**. (b)Additionally to the traditional squeeze-and-excitation mechanism that scales the feature maps by the computed coefficients, here the coefficients are passed through a softmax before scaling. This operation can be thought of as a feature map **multiplexer**.

wise convolution. By transferring the load from the pointwise convolutions to MDW convolutions our network achieves high compression rates without incurring in significant decrease in accuracy. We conjecture that our network makes better usage of the redundant feature maps produced by the pointwise convolutions. A similar observation was done by Han *et al.* [18].

Efficient, Mean and Variance Aware Channel Attention. Squeeze-and-Excitation networks [27] (SENet) was a pioneer in proposing channel attention. The purpose of the SENet is to model interactions between channels of intermediate activations and re-scale them appropriately. Their approach involves adding an MLP per convolutional layer in order to compute conditional channel-wise scalings. The scalings $\boldsymbol{s} \in \mathbb{R}^n$ are obtained from the channel-wise means $\boldsymbol{\mu} \in \mathbb{R}^n$ in the following way:

$$\boldsymbol{s} = \mathrm{MLP}(\boldsymbol{\mu}), \qquad \boldsymbol{\mu}_n = \frac{1}{f_1 f_2} \sum_{f_1} \sum_{f_2} A_{n,f_1,f_2} \tag{1}$$

ECA-Net [61] replaced this MLP with a single 1D convolutional layer parameterized by $\boldsymbol{k} \in \mathbb{R}^z$: $\boldsymbol{s} = \boldsymbol{\mu} * \boldsymbol{k}$. $z \in \mathbb{Z}^+$ is the kernel size and good results can be obtained with kernels as small as $z = 3$. For our experiments we empirically found that a kernel of size $z = 7$ provided the best results.

Given that the numbers of parameters is drastically reduced by replacing the MLP with a convolution, we augmented the ECA-Net formulation by adding the channel-wise variances $\boldsymbol{\sigma}^2 \in \mathbb{R}^n$ to the scalings computation which reuses the already computed means. We concatenate the mean and variance vectors on a new dimension and convolve with $\boldsymbol{k} \in \mathbb{R}^{z \times 2}$:

$$\boldsymbol{s} = [\boldsymbol{\mu}, \boldsymbol{\sigma}^2]_{dim=1} * \boldsymbol{k}, \qquad \boldsymbol{\sigma}_n^2 = \frac{1}{f_1 f_2} \sum_{f_1} \sum_{f_2} (A_{n,f_1,f_2} - \boldsymbol{\mu}_n)^2. \tag{2}$$

Finally, a non-linearity is applied on the computed scalars $\boldsymbol{s}$ prior to exciting the n output channels. As depicted in Figure 3 we tested both sigmoid (default in SENet) and softmax nonlinearities. The latter one is only applicable for MDW convolution layers. The softmax is implemented by first reshaping $\boldsymbol{s} \in \mathbb{R}^n$ into $\boldsymbol{s} \in \mathbb{R}^{m \times h}$ where n denotes the number of output channels, and m denotes the number of input channels, with $n = m \times h$. Then the softmax is taken across the heads dimension:

$$\boldsymbol{s}_{sig} = \text{sigmoid}(\boldsymbol{s}), \qquad \boldsymbol{s}_{sof} = \text{softmax}(\boldsymbol{s}/T)_{dim=1}. \tag{3}$$

where T is the temperature hyperparameter. If used with MDW, the sigmoid case can be though of as a channel mixer, while the softmax case can be though of as a feature map selector or multiplexer.

Including second-order information in the channel-wise attention has been previously explored in [14]; however, the authors compute a per-layer channels covariance matrix which is considerably more expensive than our approach.

Table 1: FlyNet ImageNet accuracy with different activations and channel reductions in the MDW convolutions. All the models have the same number of parameters and similar MAdds.

Model	Activation	Reduction	Top-1 Acc	Top-5 Acc
FlyNet-h3 0.5x	ECA+Sigmoid	Sum+Relu	57.6	80.5
FlyNet-h3 0.5x	ECA+Sigmoid	Funnel Max-out	58.8	80.9
FlyNet-h3 0.5x	ECA+Sigmoid	Max-out	58.9	81.0
FlyNet-h3 0.5x	ECA+Softmax(T=3)	Max-out	58.9	81.0
FlyNet-h3 0.5x	ECA+Softmax(T=10→ 1)	Max-out	**59.1**	**81.3**

In Table 1 we performed experiments on a network with $h = 3$ heads to compare the different combinations of excitation activations and feature map reductions. We can observe that Softmax with temperature annealing followed by Max-out reduction performed slightly better than the other strategies.

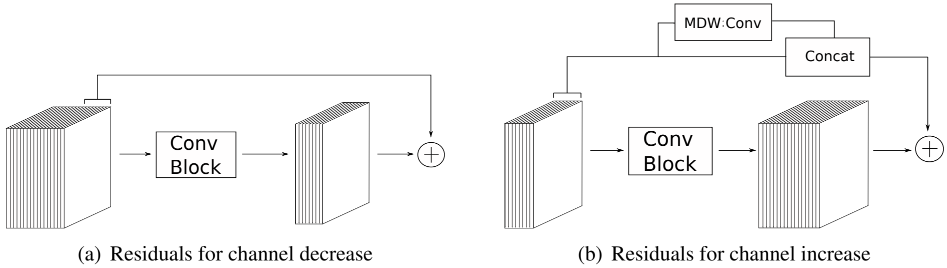
Dense Light Residuals. Unlike a traditional ResNet that uses residual connections [21, 53] as shortcuts between high dimensional representations, MobileNetV2 and V3 make use of them to connect its low dimensional features in the linear bottlenecks, and are thus named inverted residuals. Whenever there is a mismatch in the number of input and output channels, ResNet uses 1x1 convolutions to adjust the number of channels; however, this incurs in a significant number of weights and MAdds, and therefore are avoided in MobileNet by simply not placing them. Here, as depicted in Figure 4 we propose an alternative depending on whether the number of channels should increase or decrease:

$$\mathsf{R} = \begin{cases} \mathsf{A}^{l-1}_{0:n,:,:} & if \quad m > n \\ [\mathsf{A}^{l-1}, \text{MDW}(\mathsf{A}^{l-1})]_{dim=0} & if \quad m < n \end{cases}, \tag{4}$$

where R, the residual connection, now matches the dimension of A^{l-1}, and l is the layer index. MDW is used to both fill the residuals shortfall and spatial downsampling.

In the case of $m < n$, analogously to the case $m > n$, we tested only bridging the available input channels with residual connections to the first n channels, however this strategy did not provide improvements.

Following the analysis from the Sandglass block [72], we added dense residuals to the MobileNet architecture by bridging both the low dimensional representations and the high dimensional representations. As illustrated in the ablation studies detailed In Table 2, we achieved increased accuracy and generalization at practically no overhead.



(a) Residuals for channel decrease (b) Residuals for channel increase

Figure 4: Light residuals for channel mismatch. Unlike ResNet [21] which fully relies on 1x1 convolutions to adjust the number of channels in mismatching residual connections, we use a lighter version that compliments only for the channels shortfall. Both cases use Depthwise convolutions for spatial down-sampling when required.

Table 2: FlyNet ImageNet accuracy with light residual connections. See Section 3.2.

Backbone	Residual	Top-1 Acc	Top-5 Acc	Params	Madds
FlyNet-h3 0.4x	No Residual	54.5	78.0	0.657M	26.5M
FlyNet-h3 0.4x	Default in MobileNetV3	55.5	78.8	0.657M	26.5M
FlyNet-h3 0.4x	3x3 Conv	56.0	79.3	0.673M	30.9M
FlyNet-h3 0.4x	3x3 DW - PW	55.6	78.9	0.659M	27.5M
FlyNet-h3 0.4x	3x3 MDW - Concat	55.9	79.2	0.658M	27.0M

3.3 Quantization as Regularization

Quantized networks have been observed to harm training accuracy at the expense of reduction in computational resources, but present high generalization due to the highly restrictive permissible set of values, preventing them from over adjusting to the data. Therefore, we propose training with low quantization noise by manually tuning the bitwidth/resolution of the network.

Restrictive codebooks and clipping functions have been empirically observed to harm accuracy [7]. Here we use adaptive quantization, meaning that no clipping is performed and the codebook is dynamic. This is because no codebook will be used during inference; thus, it can vary across training iterations. Similarly, the quantization levels q do not have to be powers of 2.

Analogously to Dropout being modelled as multiplicative Gaussian noise on the weights [62], a q-level uniform quantization function $Q(\cdot)$ as follows:

$$Q(x) = \Delta \cdot (\text{round}(\frac{\text{x}}{\Delta})), \qquad \Delta = \frac{1}{\text{q}-1} \tag{5}$$

can be modelled as additive noise $x + \varepsilon$ with an uniform distribution $\varepsilon \sim \mathbb{U}(-\Delta/2, \Delta/2)$, where x has been re-scaled to the range $[0,1]$. Refer to Banner *et al.* [2] for details.

In order to back-propagate through the non-differentiable rounding function, we resort to the Straight Through Estimator (STE) originally proposed in [3], and employed ubiquitously across quantization works [9, 28, 47, 73]. The STE is defined as: $\frac{dL}{dx} \approx \frac{dL}{dQ(x)}$, where L denotes the loss function.

We inserted our quatization regularization as a drop-in replacement for Dropout in the

MobileNetV3 backbone. As initialization, we used a pre-trained full-precision network. We swept different quantization levels in section 4.3.

4 Experiments

4.1 ImageNet

We conducted experiments in the ImageNet ILSVRC 2012 [35] dataset which consists of 1.2M labelled images for the task of classification with 50K test images spread across 1000 classes. The images resolution used for training and computing the number of MAdds is of 224^2. For testing we first resize the smallest size of the image to 266 pixels before doing a single center crop of 224^2.

We used the momentum SGD optimizer. For the learning rate we used a cosine schedule starting at $lr = 0.4$ and momentum=0.2 during 150 epochs with mini-batch of 64. We observed that both Dropout and quantization regularizations harmed the accuracy on small networks (*e.g.* FlyNet 0.5x), therefore we only applied our regularization on models above 1.6M parameters. Similarly, weight decay was set to $1e^{-5}$ for models with #Params<1.6M and $4e^{-5}$ for the remaining ones.

Table 3: FlyNet performance results in ImageNet.

Model	Top-1 Acc	Top-5 Acc	Params	MAdds
MobileNetV3 0.15x[37]	33.7	57.2	1.0M	4M
MicroNet-M0 [37]	46.6	70.6	1.0M	4M
FlyNet-h3 0.4x	**55.9**	**79.2**	0.65M	26M
MobileNetV3 0.2x[37]	41.1	65.2	1.2M	6M
MicroNet-M1#	49.4	72.9	1.2M	5M
MicroNet-M1	51.4	74.5	1.8M	6M
EfficientNet-B [56]	56.7	79.8	1.3M	24M
FlyNet-h3 0.5x	**59.1**	**81.3**	0.86M	34M
MobileNetV3 0.35x+BFT [59]	55.2	-	1.4M	15M
MobileNetV3 0.5x [24]	58.0	-	1.6M	21M
MicroNet-M2#	58.2	80.1	1.4M	11M
MicroNet-M2	59.4	80.9	2.4M	12M
TinyNet-E [19]	59.9	81.8	2.0M	24M
ShuffleNetV2 0.5x [42]	60.3	-	1.4M	41M
FlyNet-h3 0.6x	**61.5**	**83.4**	1M	46M
ShuffleNetV2 0.5x+BFT [59]	61.3	-	1.4M	41M
MicroNet-M3#	61.3	82.9	1.6M	20M
FlyNet-h3 0.7x	**63.3**	**84.5**	1.3M	54M
MicroNet-M3	62.5	83.1	2.6M	21M
Mobile-Former-26M [5]	64.0	-	3.2M	26M
EtinyNet [66]	65.5	86.2	0.98M	117M
FlyNet-h3 0.8x	**65.7**	**86.2**	1.6M	66M

The results of FlyNet on ImageNet are listed in Table 3. We can see that FlyNet is very efficient towards the extremely low parameter regime, with FlyNet-h3 0.4x (3 heads and width multiplier of 0.4) widely outperfoming MobileNetV3 0.15x and MicroNet-M0.

FlyNet effectively reduces the number of parameters while maintaining a reasonable number of MAdds in comparison to EtinyNet [66], since both memory accesses and operations can be expensive in terms of energy consumption as discussed in [23] and [9].

4.2 COCO Object Detection

Table 4: FlyNet performance in COCO object detection. We reported the number of parameters and MAdds of the backbones used as drop-in replacement.

Backbone	DET Framework	Params	MAdds	mAP
FlyNet-h3 0.4x	RCNN	0.14M	26M	21.9
FlyNet-h3 0.5x		0.19M	34M	23.2
MicroNet-M2		0.58M	12M	22.7
FlyNet-h3 0.6x		0.24M	46M	24.4
MobileNetV3 1.0x		0.89M	56M	25.9
MicroNet-M3		0.69M	21M	26.2
FlyNet-h3 0.8x		0.41M	86M	**27.0**
FlyNet-h3 0.4x	RetinaNet	0.14M	26M	22.9
MicroNet-M2		0.58M	12M	22.6
FlyNet-h3 0.5x		0.19M	34M	23.7
MobileNetV3 1.0x		0.89M	56M	24.0
FlyNet-h3 0.6x		0.24M	46M	24.6
MicroNet-M3		0.69M	21M	25.4
FlyNet-h3 0.8x		0.41M	86M	**27.2**

COCO [39] object detection is large-scale dataset consisting of 80K training and 40K validation images with annotated boxes for 90 different classes.

We implemented our FlyNet backbone in the Faster-RCNN and RetinaNet frameworks compatible with MobileNetV3 without any modifications. We used pretrained backbones and fine-tuned for 26 epochs with multi-step learning rate starting at $2.5e^{-3}$, multiplied by a factor of 0.1 at epochs 16 and 22. We set weight decay to $4e^{-5}$ and momentum 0.9.

We reported Average Precision ($AP^{0.5}$) as well as the number of parameters and MAdds (computed using 224×224 images) of the backbones (ignoring the heads) used as drop-in replacement. FlyNet outperforms both MicroNet and MobileNetV3 as reported in [37] in terms of number of parameters. In the low parameter regime, FlyNet-h3 0.5x performs almost 3 times as many MAdds than MicroNet-M2 but is 3 times smaller with higher accuracy.

4.3 Ablation Studies

In this section we test the individual contribution of our ideas. Some ablations have been included in the corresponding sections in the main body of the paper.

Number of MDW convolution heads. In Table 5 we investigate the impact of the additional heads in the MDW convolutions. We tested with heads in the range $[1,5]$. We observed a monotonic increase in accuracy for all of them, with a total gain of more than 3% from its single head counterpart at almost no overhead in parameters and less than twice the operations. We also observed that the largest accuracy jump comes from the first additional head at very little cost.

Quantization levels. As common convention, we initilized the quantized networks with a pretrained full-precision network, and we set the momentum to 0.9. We did not test training from scratch. As previously mentioned, we found regularizing small networks harms their performance; therefore we tested our regularization on the largest version of FlyNet (1.0x).

Table 5: ImageNet accuracy for different number of MDW heads.

Model	Heads	Top-1 Acc	Top-5 Acc	Params	MAdds
FlyNet 0.5x	1	56.7	79.6	0.80M	24M
FlyNet 0.5x	2	58.2	81.0	0.83M	29M
FlyNet 0.5x	3	59.1	81.4	0.86M	34M
FlyNet 0.5x	4	59.8	81.9	0.89M	39M
FlyNet 0.5x	5	**59.9**	**82.0**	0.92M	44M

Table 6: ImageNet accuracy for different quantization levels (q-levels). See Section 3.3

Model	q-levels	Top-1 Acc	Top-5 Acc
FlyNet-h3 1.0x	2^{32}	67.2	87.5
FlyNet-h3 1.0x	16	67.3	87.7
FlyNet-h3 1.0x	8	**67.4**	**87.8**
FlyNet-h3 1.0x	6	67.1	87.4
FlyNet-h3 1.0x	4	66.1	87.1

5 Conclusions

We developed a series of simple but effective architectural modifications that can be integrated into any neural architecture to provide accuracy boosts at very little overhead. Particularly, our contributions are aimed at compressed networks in the extremely low parameter regime (sub 1M). We leveraged a MobileNetV3 backbone to devise the FlyNet architecture. We performed ablation studies and experiments on different large-scale benchmarks to analyze the impact of our contributions. We believe our research will set grounds for further automatic architecture search aimed at mobile and tiny machine learning.

References

[1] Colby Banbury, Chuteng Zhou, Igor Fedorov, Ramon Matas, Urmish Thakker, Dibakar Gope, Vijay Janapa Reddi, Matthew Mattina, and Paul Whatmough. Micronets: Neural network architectures for deploying tinyml applications on commodity microcontrollers. *Proceedings of Machine Learning and Systems*, 3:517–532, 2021.

[2] Ron Banner, Itay Hubara, Elad Hoffer, and Daniel Soudry. Scalable methods for 8-bit training of neural networks. *arXiv preprint arXiv:1805.11046*, 2018.

[3] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[4] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11030–11039, 2020.

[5] Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Xiaoyi Dong, Lu Yuan, and Zicheng Liu. Mobile-former: Bridging mobilenet and transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5270–5279, 2022.

[6] Yunpeng Chen, Yannis Kalantidis, Jianshu Li, Shuicheng Yan, and Jiashi Feng. A^2-nets: Double attention networks. *arXiv preprint arXiv:1810.11579*, 2018.

[7] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

[8] Francois Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 1251–1258, 2017.

[9] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 3123–3131, 2015.

[10] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[11] Hassan Dbouk and Naresh Shanbhag. Generalized depthwise-separable convolutions for adversarially robust and efficient neural networks. *Advances in Neural Information Processing Systems*, 34:12027–12039, 2021.

[12] Emily L Denton, Wojciech Zaremba, Joan Bruna, Yann LeCun, and Rob Fergus. Exploiting linear structure within convolutional networks for efficient evaluation. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 1269–1277, 2014.

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

[14] Zilin Gao, Jiangtao Xie, Qilong Wang, and Peihua Li. Global second-order pooling convolutional networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3024–3033, 2019.

[15] Ian Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR, 2013.

[16] Sam Gross, Marc'Aurelio Ranzato, and Arthur Szlam. Hard mixtures of experts for large scale weakly supervised vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6865–6873, 2017.

[17] Jianyuan Guo, Kai Han, Han Wu, Chang Xu, Yehui Tang, Chunjing Xu, and Yunhe Wang. Cmt: Convolutional neural networks meet vision transformers. *arXiv preprint arXiv:2107.06263*, 2021.

[18] Kai Han, Yunhe Wang, Qi Tian, Jianyuan Guo, Chunjing Xu, and Chang Xu. Ghostnet: More features from cheap operations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1580–1589, 2020.

[19] Kai Han, Yunhe Wang, Qiulin Zhang, Wei Zhang, Chunjing Xu, and Tong Zhang. Model rubik's cube: Twisting resolution, depth and width for tinynets. *Advances in Neural Information Processing Systems*, 33:19353–19364, 2020.

[20] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *Proc. Int. Conf. Learn. Repren.*, 2016.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 770–778, 2016.

[22] Yang He, Ping Liu, Ziwei Wang, and Yi Yang. Pruning filter via geometric median for deep convolutional neural networks acceleration. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2019.

[23] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC)*, pages 10–14. IEEE, 2014.

[24] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324, 2019.

[25] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[26] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. Gather-excite: Exploiting feature context in convolutional neural networks. *arXiv preprint arXiv:1810.12348*, 2018.

[27] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018.

[28] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 4107–4115, 2016.

[29] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[30] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. Int. Conf. Mach. Learn.*, pages 448–456, 2015.

[31] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning*, pages 4651–4664. PMLR, 2021.

[32] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

[33] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[34] Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28:2575–2583, 2015.

[35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 1097–1105, 2012.

[36] Yann LeCun, John Denker, and Sara Solla. Optimal brain damage. *Advances in neural information processing systems*, 2, 1989.

[37] Yunsheng Li, Yinpeng Chen, Xiyang Dai, Dongdong Chen, Mengchen Liu, Lu Yuan, Zicheng Liu, Lei Zhang, and Nuno Vasconcelos. Micronet: Improving image recognition with extremely low flops. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 468–477, 2021.

[38] Ji Lin, Wei-Ming Chen, Han Cai, Chuang Gan, and Song Han. Mcunetv2: Memory-efficient patch-based inference for tiny deep learning. *arXiv preprint arXiv:2110.15352*, 2021.

[39] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[40] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *Proc. Int. Conf. Learn. Repren.*, 2019.

[41] Christos Louizos, Max Welling, and Diederik P Kingma. Learning sparse neural networks through *l*_0 regularization. In *Proc. Int. Conf. Learn. Repren.*, 2018.

[42] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

[43] Ningning Ma, Xiangyu Zhang, and Jian Sun. Funnel activation for visual recognition. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 351–368. Springer, 2020.

[44] Xuezhe Ma, Xiang Kong, Sinong Wang, Chunting Zhou, Jonathan May, Hao Ma, and Luke Zettlemoyer. Luna: Linear unified nested attention. *Advances in Neural Information Processing Systems*, 34, 2021.

[45] Ravi Teja Mullapudi, William R Mark, Noam Shazeer, and Kayvon Fatahalian. Hydranets: Specialized dynamic architectures for efficient inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8080–8089, 2018.

[46] Jongchan Park, Sanghyun Woo, Joon-Young Lee, and In So Kweon. Bam: Bottleneck attention module. *arXiv preprint arXiv:1807.06514*, 2018.

[47] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *Proc. Eur. Conf. Comp. Vis.*, pages 525–542, 2016.

[48] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[49] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 4510–4520, 2018.

[50] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[51] Laurent Sifre and Prof Stéphane Mallat. Rigid-motion scattering for image classification author. *English. Supervisor: Prof. Stéphane Mallat. Ph. D. Thesis. Ecole Polytechnique*, 2014.

[52] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[53] Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. Highway networks. *arXiv preprint arXiv:1505.00387*, 2015.

[54] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11166–11175, 2019.

[55] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proc. AAAI Conf. on Arti. Intel.*, volume 4, page 12, 2017.

[56] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.

[57] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34, 2021.

[58] Asher Trockman and J Zico Kolter. Patches are all you need? *arXiv preprint arXiv:2201.09792*, 2022.

[59] Keivan Alizadeh Vahid, Anish Prabhu, Ali Farhadi, and Mohammad Rastegari. Butterfly transform: An efficient fft based neural architecture design. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12021–12030. IEEE, 2020.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[61] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: efficient channel attention for deep convolutional neural networks, 2020 ieee. In *CVF Conference on Computer Vision and Pattern Recognition (CVPR). IEEE*, 2020.

[62] Sida Wang and Christopher Manning. Fast dropout training. In *international conference on machine learning*, pages 118–126. PMLR, 2013.

[63] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.

[64] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 4376–4382. IEEE, 2019.

[65] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, pages 5987–5995, 2017.

[66] Kunran Xu, Yishi Li, Huawei Zhang, Rui Lai, and Lin Gu. Etinynet: Extremely tiny network for tinyml. 2022.

[67] Brandon Yang, Gabriel Bender, Quoc V Le, and Jiquan Ngiam. Condconv: Conditionally parameterized convolutions for efficient inference. *arXiv preprint arXiv:1904.04971*, 2019.

[68] Hongwei Yong, Jianqiang Huang, Deyu Meng, Xiansheng Hua, and Lei Zhang. Momentum batch normalization for deep learning with small batch size. In *European Conference on Computer Vision*, pages 224–240. Springer, 2020.

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

[70] Xiangyu Zhang, Jianhua Zou, Kaiming He, and Jian Sun. Accelerating very deep convolutional networks for classification and detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 38(10):1943–1955, 2016.

[71] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. *arXiv preprint arXiv:1707.01083*, 2017.

[72] Daquan Zhou, Qibin Hou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. Rethinking bottleneck structure for efficient mobile network design. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 680–697. Springer, 2020.

[73] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[74] Zhuangwei Zhuang, Mingkui Tan, Bohan Zhuang, Jing Liu, Yong Guo, Qingyao Wu, Junzhou Huang, and Jinhui Zhu. Discrimination-aware channel pruning for deep neural networks. In *Proc. Adv. Neural Inf. Process. Syst.*, 2018.

[75] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *Proc. Int. Conf. Learn. Repren.*, 2017.

[76] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proc. IEEE Conf. Comp. Vis. Patt. Recogn.*, 2018.