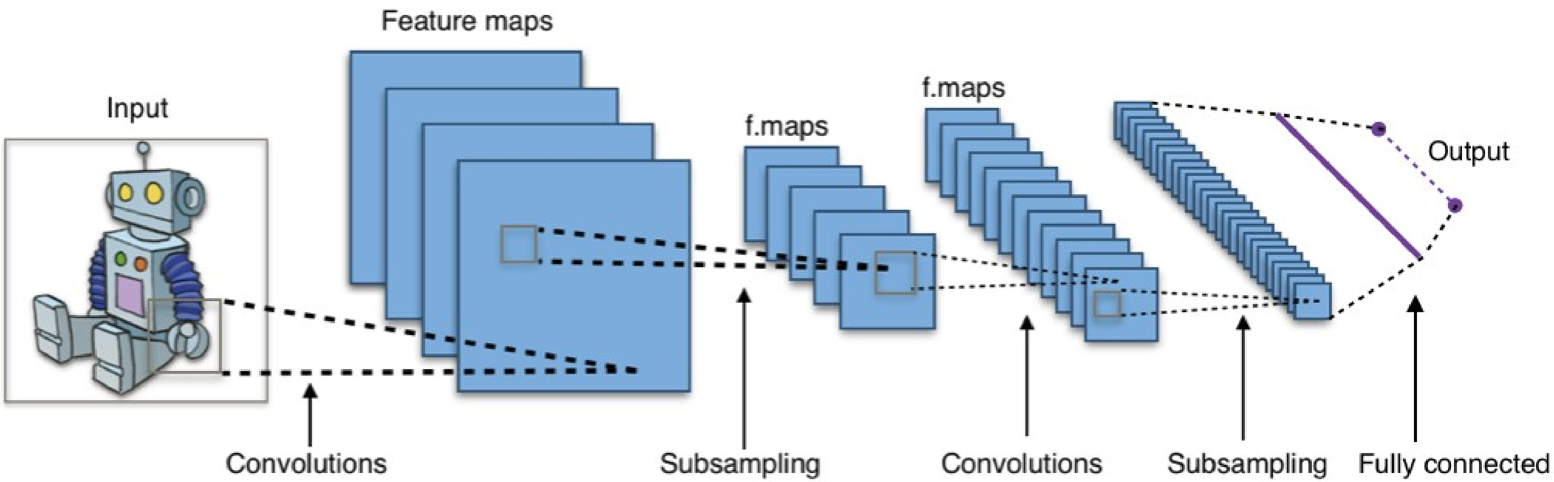
Mobilenet

Adam Jabłonowski
KNUM CORE #3 29.04.2020

What is convolutional neural network

- In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to analyzing visual imagery.



- It consists of few blocks made of combination of convolutional layers and functions adding non-linearity and fully connected layer at end of network.

What is Mobilenet

- Mobilenet is a convolutional neural network used for tasks as image classification, detetion in segmentation.
- Mobilenet is desinged to run in constrained environments like mobile phones and focuses on low inference latency
- Network is often used as element of bigger architectures.
  - For example as encoder in image segmentation framework Deeplab
    - https://github.com/tensorflow/models/tree/master/research/deeplab
    - https://arxiv.org/abs/1802.02611
- Competes with other architectures like:
  - SqueezeNet, MnasNet, BlazeFace, TinyYOLO / Darknet, SqueezeNext, ShuffleNet, CondenseNet, ESPNet, DiCENet, FBNet & ChamNet, GhostNet, MixNet, EfficientNet

Scientific papers

- „MobileNets: Efficient Convolutional Neural Networks for Mobile VisionApplications” 2017
  - https://arxiv.org/pdf/1704.04861.pdf
- „MobileNetV2: Inverted Residuals and Linear Bottlenecks” 2018
  - https://arxiv.org/pdf/1801.04381.pdf
- „MnasNet: Platform-Aware Neural Architecture Search for Mobile” 2019
  - https://arxiv.org/abs/1807.11626
- „Searching for MobileNetV3” 2019
  - https://arxiv.org/pdf/1905.02244.pdf

Previous work

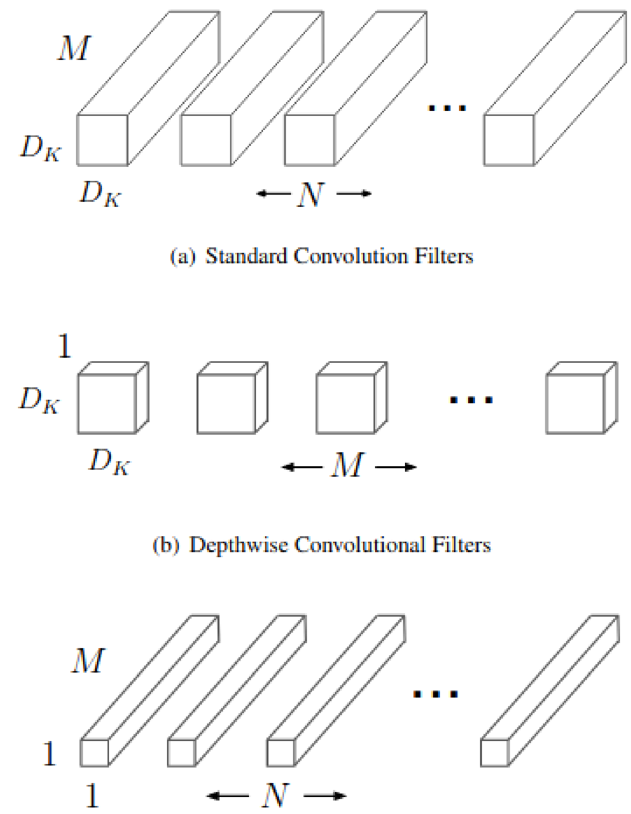
- A network out of fully factorized convolutions, showed the potential of extremely factorized networks.
  - „Flattened Convolutional Neural Networks for Feedforward Acceleration” https://arxiv.org/abs/1412.5474
- Squeezenet which uses a bottleneck approach.
  - „SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size” https://arxiv.org/abs/1602.07360
- Other reduced computation networks include:
  - „Structured Transforms for Small-Footprint Deep Learning” https://arxiv.org/abs/1510.01722
  - „Deep Fried Convnets” https://arxiv.org/abs/1412.7149

Popular methods for making network small

- Different approaches for obtaining small networks are:
  - shrinking, pruning, which cuts of some elements like neurons or neuron connections
  - factorizing, which replace one element with few simpler
  - compressing pretrained networks based on:
    - product quantization
    - hashing
    - pruning
    - vector quantization
    - Huffman coding
  - distillation, which uses a larger network to teach a smaller network.
  - low bit networks, which elements are represented by very small numbers of bits

Mobilenet v1 Depthwise separable convolutions

- Factorize a standard convolution into a depthwise convolution and a 1×1 convolution called a pointwise convolution, what drasticallly reduses computation costs.
- Assume **h** is height, **w** width, **d** depth, **i**, **j** index of layer, **k** kernel
  - Standard convolutions have the computational cost of:
    - **hi·wi·di·dj·k·k**
  - Depthwise separable convolutions cost:
    - **hi·wi·di(k·k+dj)**
- The trade off is that such kernel is less expressive



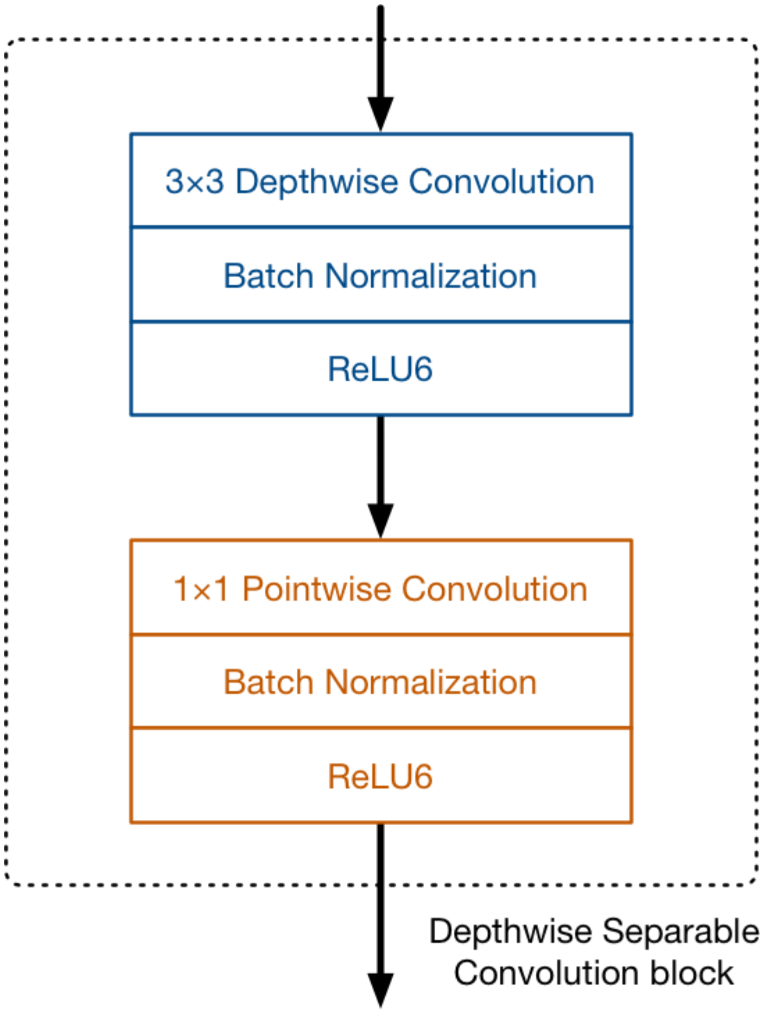
(a) Standard Convolution Filters

(b) Depthwise Convolutional Filters

(c) 1 × 1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

Mobilenet v1 Architecture

- Streamlined architecture
- All layers are followed by a batchnorm and ReLU nonlinearity with the exceptionof the final fully connected layer which has no nonlinearity and feeds into a softmax layer for classification.
  - „Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift” https://arxiv.org/abs/1502.03167
- A final average pooling reduces the spatial resolution to 1 before the fully connected layer.
- Counting depthwise and pointwise convolutions as separate layers, MobileNet has 28 layers.



Mobilenet v1 Computations

- Model structure puts nearly all of the computation into dense 1×1 convolutions. This can be implemented with highly optimized general matrix multiply (GEMM) functions. And 1×1 convolutions do not require an initial reordering in memory in order to map it to a GEMM.
- MobileNet spends 95% of it's computation time in1×1 convolutions which also has 75% of the parameters. Nearly all of the additional parameters are in the fully connected layer.

Mobilenet v1 Hyper-Parameters

- There are introduced two simple global hyper-parameters that efficiently trade off between latency and accuracy.
- width multiplier **α**
  - The number of input channels in depthwise separable convolutions **di** becomes **αdi** and the number of output channels **dj** becomes **αdj**.
  - The computational cost with width multiplier **α** is
    - **hi·wi·α·di(k·k+α·dj)**
- resolution multiplier **ρ**
  - The input image and the internal representation of every layer is subsequently reduced by the same multiplier. In practice we implicitly set **ρ** by setting the input resolution.
  - The computational cost with is
    - **ρ·hi·ρ·wi·α·di(k·k+α·dj)**
- Resolution multiplier has the effect of quadratically reducing computational cost and width multiplier roughly quadratically.

Mobilenet v1 Results

- Performance compared to other popular models on ImageNet http://image-net.org/download classification and face recognition

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0 MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

Table 9. Smaller MobileNet Comparison to Popular Models

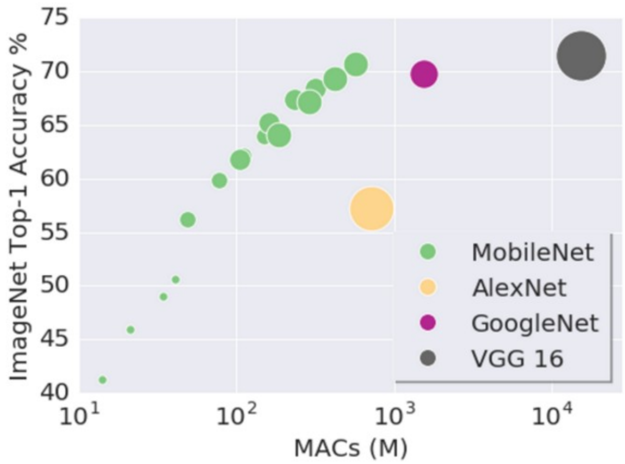
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

Table 10. MobileNet for Stanford Dogs

Model	Top-1 Accuracy	Million Mult-Adds	Million Parameters
Inception V3 [18]	84%	5000	23.2
1.0 MobileNet-224	83.3%	569	3.3
0.75 MobileNet-224	81.9%	325	1.9
1.0 MobileNet-192	81.9%	418	3.3
0.75 MobileNet-192	80.5%	239	1.9

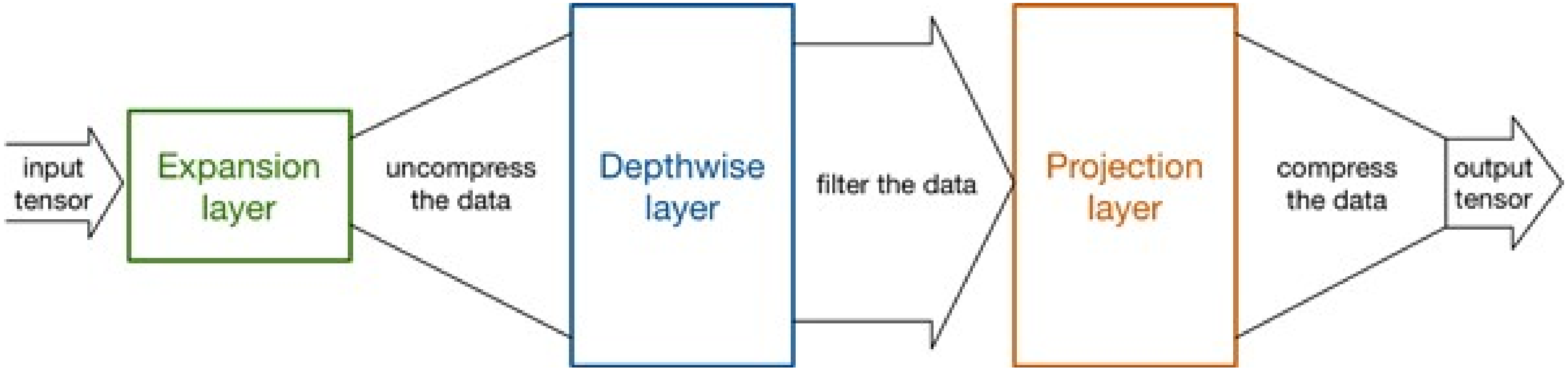
Table 14. MobileNet Distilled from FaceNet

Model	1e-4 Accuracy	Million Mult-Adds	Million Parameters
FaceNet [25]	83%	1600	7.5
1.0 MobileNet-160	79.4%	286	4.9
1.0 MobileNet-128	78.3%	185	5.5
0.75 MobileNet-128	75.2%	166	3.4
0.75 MobileNet-128	72.5%	108	3.8



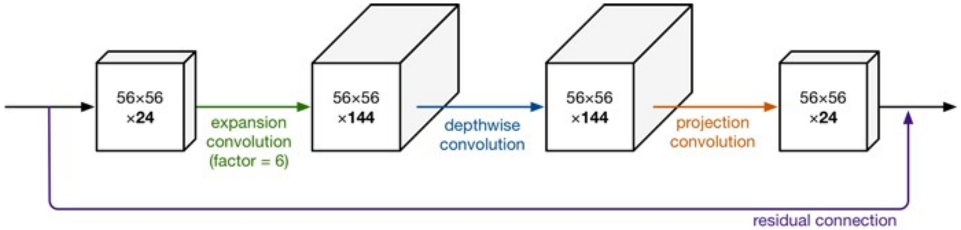
Mobilenet v2 Ideas

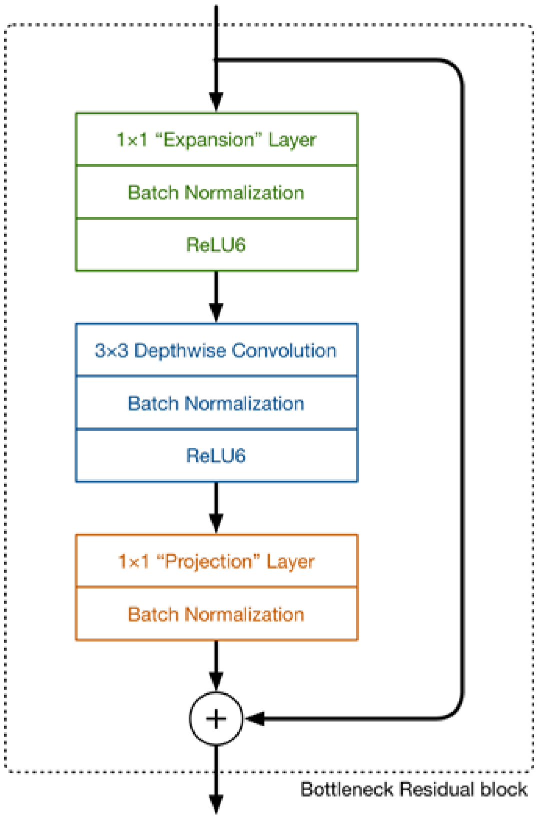
- The intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity.
- It is important to remove non-linearities in the narrow layers in order to maintain representational power.
- Approach allows decoupling of the input/output domains from the expressiveness of the transformation, which provides a convenient framework for further analysis.
- Network significantly decreases the number of operations and memory needed while retaining the same accuracy.



Mobilenet v2 Inverted residual with linear bottleneck

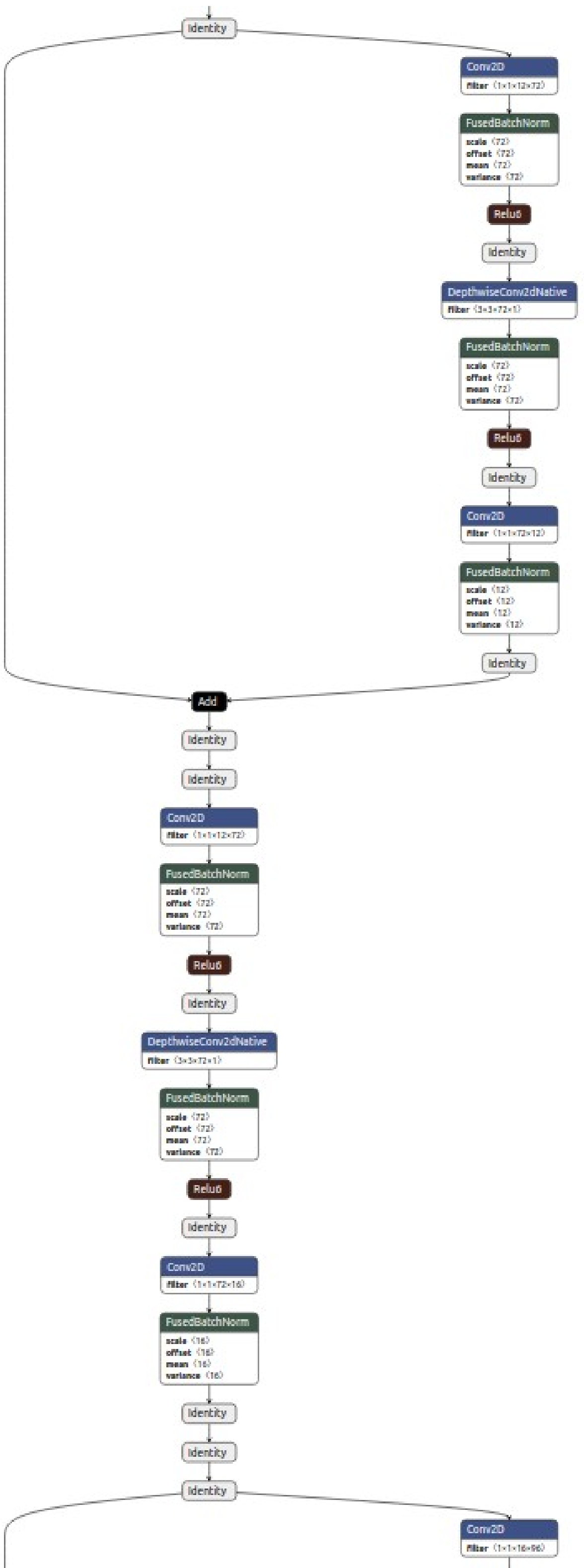
- Main improvement is a novel layer module the inverted residual with linear bottleneck.
- This module takes as an input a low-dimensional compressed representation which is
  - first expanded to high dimension
  - filtered with a lightweight depthwise convolution
  - Features are subsequently projected back to a low-dimensional representation with a linear convolution.





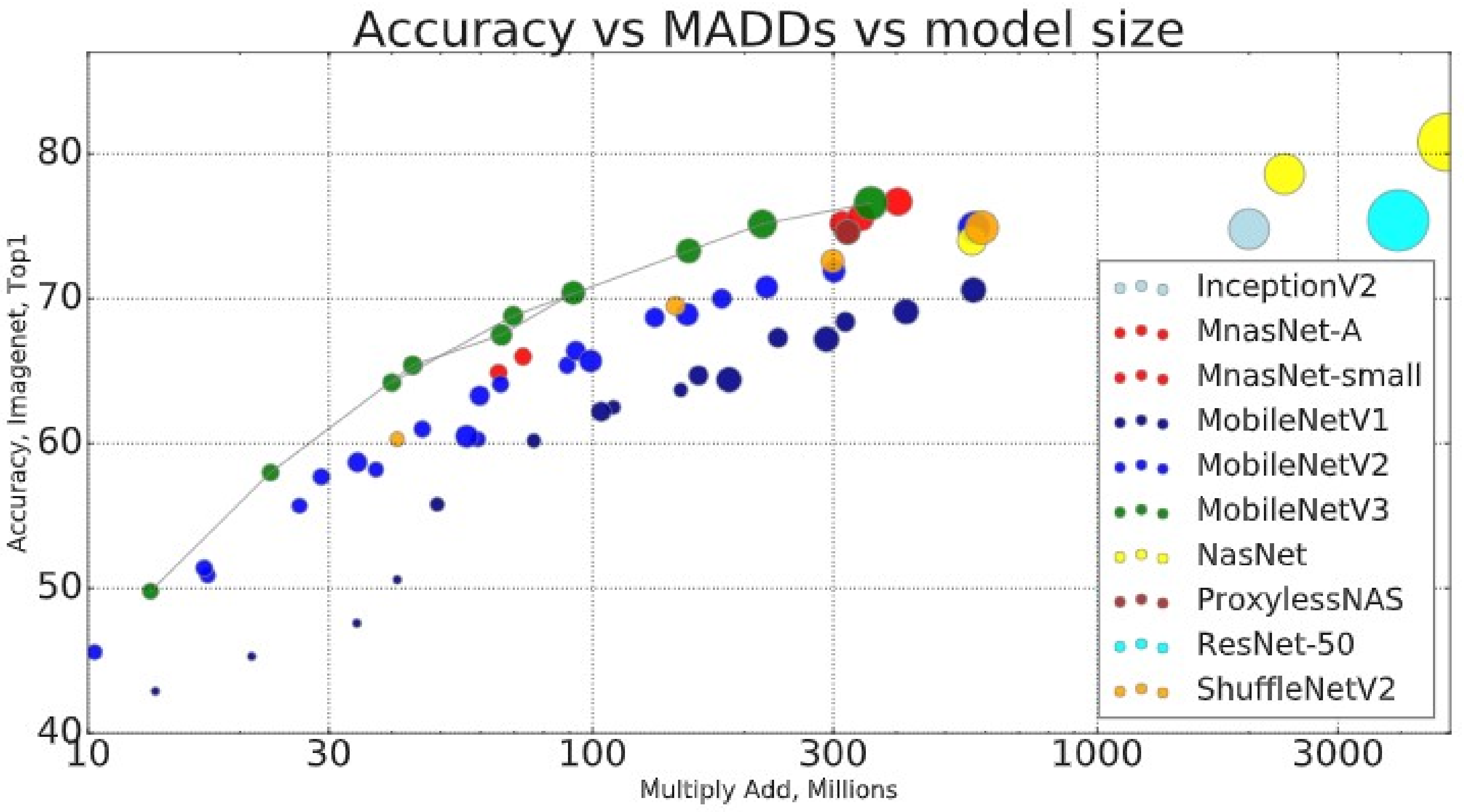
Mobilenet v2 Implementation

- The official implementation is available as part of TensorFlow-Slim model library
  - https://github.com/tensorflow/models/tree/master/research/slim/nets/mobilenet
- Module is suitable for mobile designs, because it significantly reduces the memory footprint needed during inference by never fully materializing large intermediate tensors.
- This reduces the need for main memory access in many embedded hardware designs, that provide small amounts of very fast software controlled cache memory.



Mobilenet v2 Results

- Accuracy on ImageNet



Mobilenet v2 Segmentation results

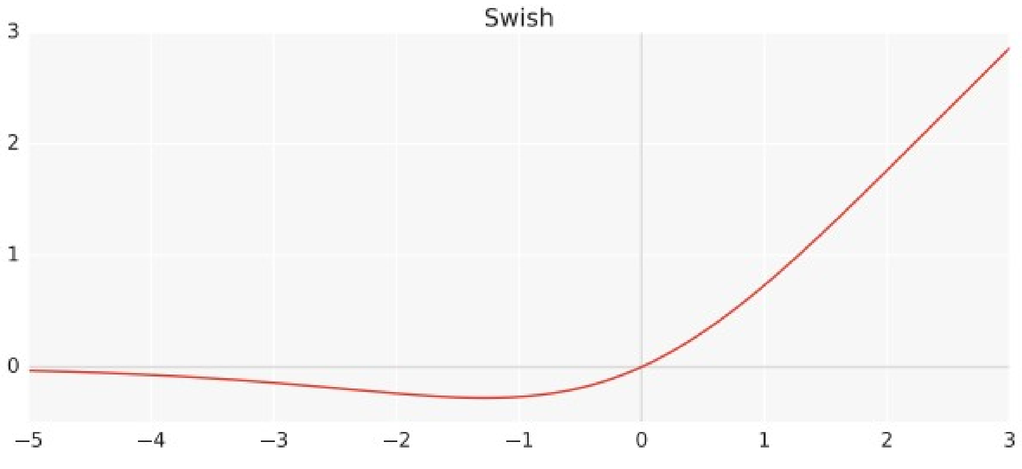
- When used with Deeplab for image segmentation, compared with state of art in PASCAL VOC 2012 segmentation test Xception65 model

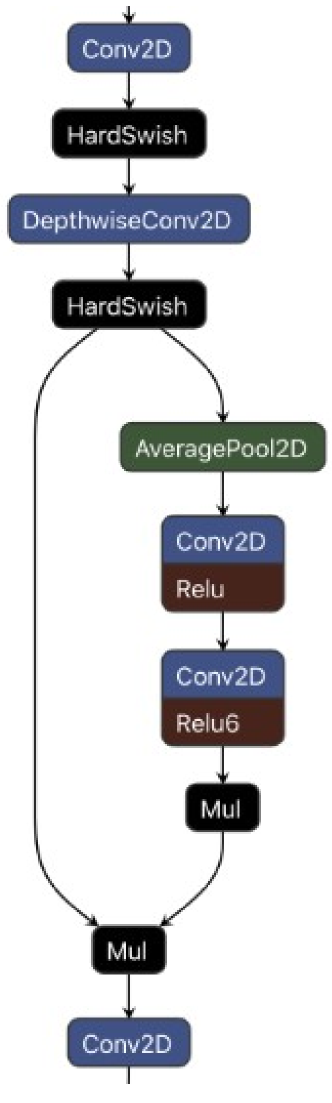
Checkpoint name	Eval OS	Eval scales	Left-right Flip	Multiply-Adds	Runtime (sec)	PASCAL mIOU	File Size
mobilenetv2_dm05_coco_voc_trainaug	16	[1.0]	No	0.88B	-	70.19% (val)	7.6MB
mobilenetv2_dm05_coco_voc_trainval	8	[1.0]	No	2.84B	-	71.83% (test)	7.6MB
mobilenetv2_coco_voc_trainaug	16 8	[1.0] [0.5:0.25:1.75]	No Yes	2.75B 152.59B	0.1 26.9	75.32% (val) 77.33 (val)	23MB
mobilenetv2_coco_voc_trainval	8	[0.5:0.25:1.75]	Yes	152.59B	26.9	80.25% **(test)**	23MB
xception65_coco_voc_trainaug	16 8	[1.0] [0.5:0.25:1.75]	No Yes	54.17B 3055.35B	0.7 223.2	82.20% (val) 83.58% (val)	439MB
xception65_coco_voc_trainval	8	[0.5:0.25:1.75]	Yes	3055.35B	223.2	87.80% **(test)**	439MB

  - „dm” means depth multiplier, „OS” means output stride (input/output ratio)
  - Models where pretrained on ImageNet then trained on coco and voc datasets

Mobilenet v3 Main changes

- The architecture was partially found through automated network architecture search (NAS)
- Inspired by MnasNet architecture found by NAS
- The main changes are
  - expensive layers were redesigned
  - use of Swish instead of ReLU6
  - squeeze-and-excitation modules





MnasNet

- Model found by network architecture search
  - „MnasNet: Platform-Aware Neural Architecture Search for Mobile” https://arxiv.org/abs/1807.11626
- MnasNet was built upon the MobileNetV2 structure by introducing lightweight attention modules based on squeeze and excitation into the bottleneck structure.
- Squeeze-and-Excitation (SE) block adaptively recalibrates channel-wise feature responses by explicitly modelling interdependencies between channels.

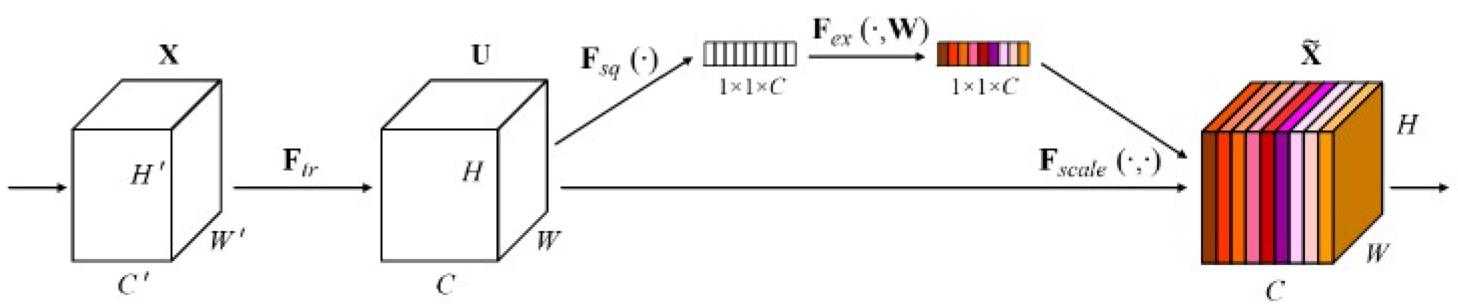


Fig. 1. A Squeeze-and-Excitation block.

MnasNet Sqeeze and excitation

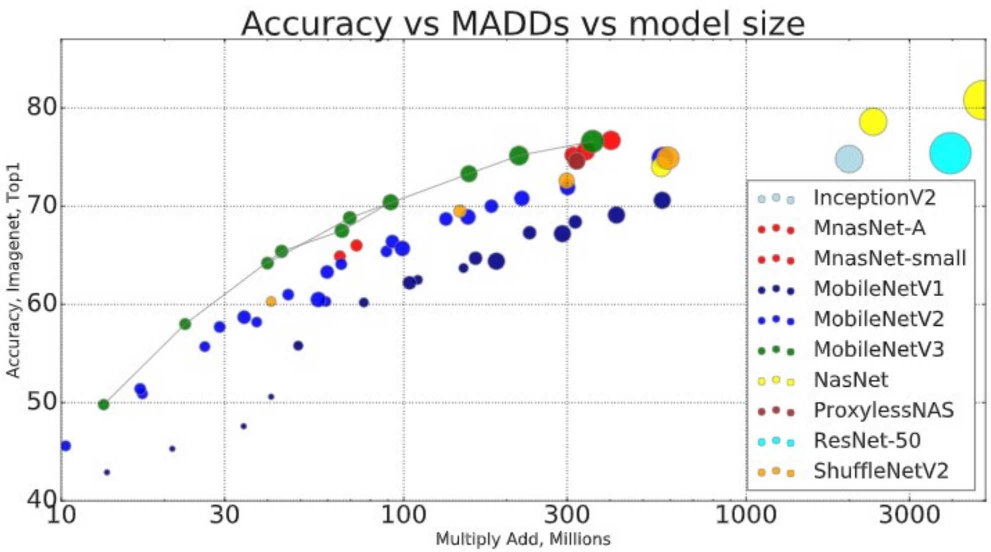
- squeeze
  - Squeeze global spatial information into a channel descriptor
  - Use global average pooling to generate channel-wise statistics
  - More sophisticated strategies,then the simplest aggregation technique, global average pooling, could be employed
- excitation
  - To make use of the information aggregated in the squeeze operation, follow it with a second operation, which aims to fully capture channel-wise dependencies
  - Employ a simple gating mechanism with a sigmoid activation, parameterised by forming a bottleneck with two fully-connected (FC) layers around the non-linearity
  - The final output of the block is obtained by rescaling with the activations
- SE blocks intrinsically introduce dynamics conditioned on the input, resulting in self-attention function on channels, whose relationships are not confined to the local receptive field the convolutional filters.

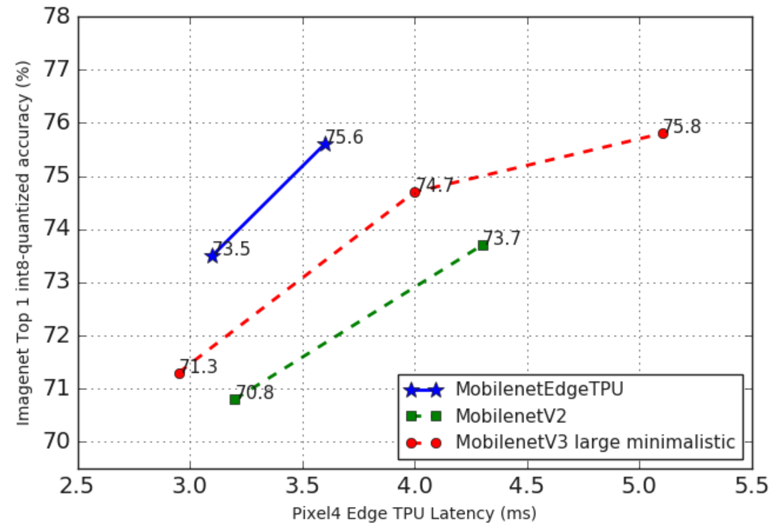
Mobilenet v3 vs MnasNet

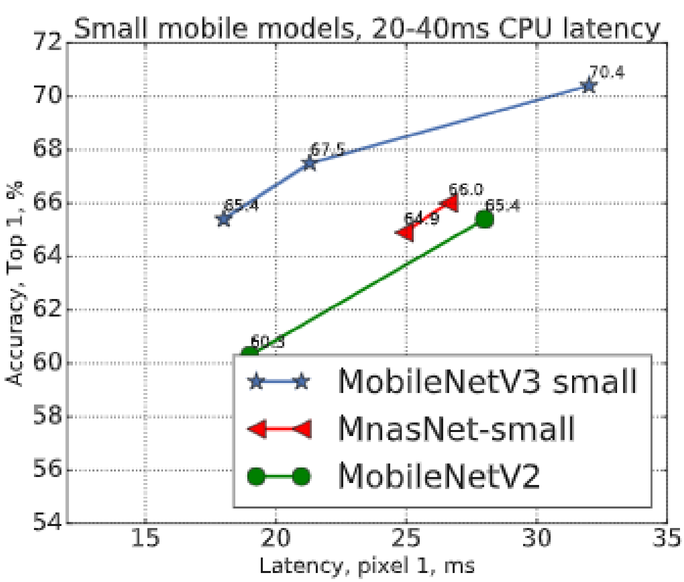
- Differences with MnasNet
  - the activation is h-swish (but as mentioned in the earlier layers it is ReLU)
  - the number of filters used by the expansion layers is different (the NetAdapt algorithm was used to find optimal values for these)
  - the number of channels that are output by the bottleneck layers may be different (also found by NetAdapt)
  - the squeeze-and-excitation (SE) modules only reduce the number of channels by a factor 3 or 4
  - instead of a sigmoid, the SE module uses the formula ReLU6(x + 3) / 6 as a rough approximation (just like what h-swish does)

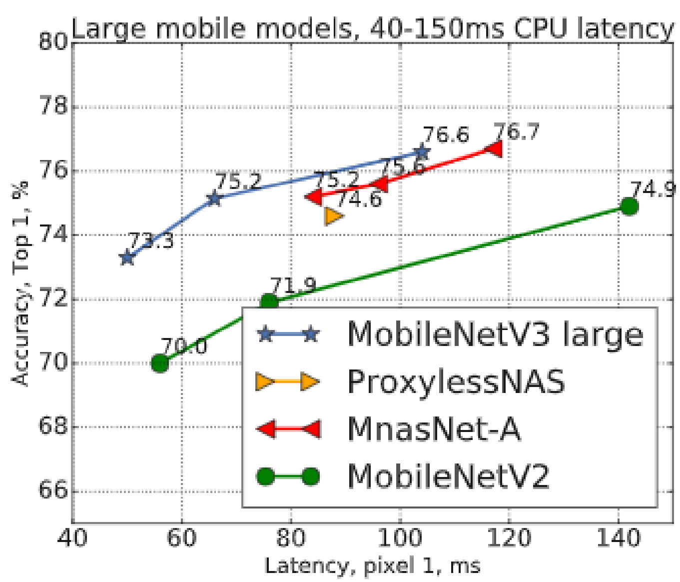
Mobilenet v3 Results

- Accuracy on ImageNet and latency









Deploying model

- U can use neural network inference frameworks like
  - TensorFlow Lite
  - Mace
- quantize your pretrained model to speed up CPU inference
  - Quantization replaces float32 model variables used while training with uint8 values, which slightly decreases quality of network output
- use quantization aware training

References

- Papers
  - „MobileNets: Efficient Convolutional Neural Networks for Mobile VisionApplications” https://arxiv.org/pdf/1704.04861.pdf
  - „MobileNetV2: Inverted Residuals and Linear Bottlenecks” https://arxiv.org/pdf/1801.04381.pdf
  - „MnasNet: Platform-Aware Neural Architecture Search for Mobile” https://arxiv.org/abs/1807.11626
  - „Searching for MobileNetV3” https://arxiv.org/pdf/1905.02244.pdf
- Images
  - https://machinethink.net/blog/mobilenet-v2/
  - https://medium.com/@neuralnets/swish-activation-function-by-google-53e1ea86f820
  - Netron model visualizer https://lutzroeder.github.io/netron/
  - PASCAL VOC 2012

End