**Sensorfusion:**

D. W. Fincher and D. F. Mix, “[Multi-sensor data fusion using neural networks](http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=142240),” 1990, pp. 835–838.

[On-line prediction of surface finish and dimensional deviation in turning using neural network based sensor fusion](https://www.sciencedirect.com/science/article/pii/S0890695597000138/pdf?md5=fdf6229e1d60702045b3aacc4692c3ca&pid=1-s2.0-S0890695597000138-main.pdf)

[End-to-End Training of Deep Visuomotor Policies](https://arxiv.org/pdf/1504.00702.pdf)

[Maximum likelihood neural networks for sensor fusion and adaptive classification](https://ac.els-cdn.com/0893608091900354/1-s2.0-0893608091900354-main.pdf?_tid=1391cb49-7a70-4b1b-a986-29f141361bae&acdnat=1520848086_cab4990d6cf23f78a7052cf7b73d5c1d)

[Multi-sensor fusion in body sensor networks: State-of-the-art and research challenges](https://ac.els-cdn.com/S156625351630077X/1-s2.0-S156625351630077X-main.pdf?_tid=4d1c5bbc-e428-4108-aaac-bf3eb8160bad&acdnat=1520848231_5842ed36418701e351d04c5b821e3c26)

[An Introduction to Sensor Fusion](https://www.researchgate.net/profile/Wilfried_Elmenreich/publication/267771481_An_Introduction_to_Sensor_Fusion/links/55d2e45908ae0a3417222dd9/An-Introduction-to-Sensor-Fusion.pdf)

B. Khaleghi, A. Khamis, F. O. Karray, and S. N. Razavi, “[Multisensor data fusion: A review of the state-of-the-art](https://pdfs.semanticscholar.org/a1a1/57018e44474b3c4ff776f734e89e033f20d2.pdf),” Information Fusion, vol. 14, no. 1, pp. 28–44, Jan. 2013.

**ANN Structures:**

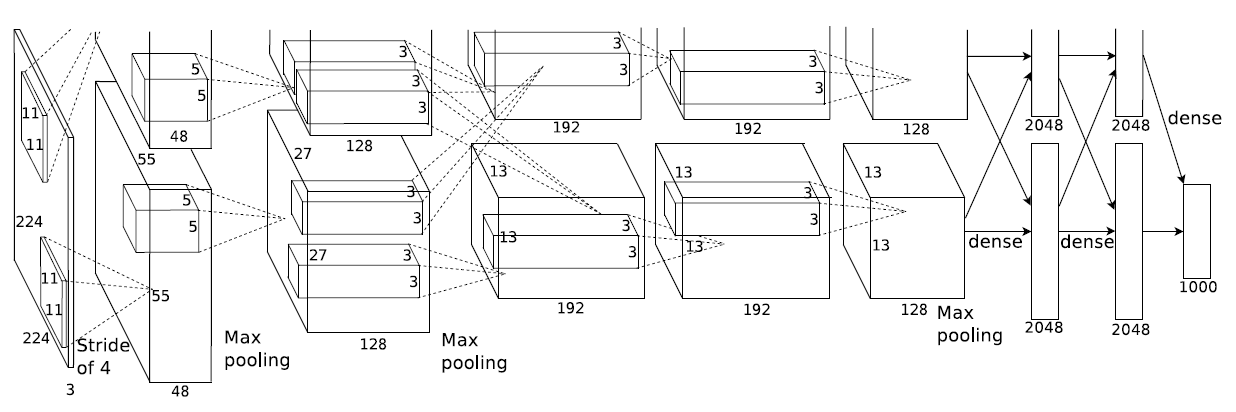
1. **AlexNet (2012)** champion of ILSVRC-2012

**Original paper:** [ImageNet Classification with Deep Convolutional Neural Networks](https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf)

**Dataset:** ImageNet over 15 million labeled high-resolution images belonging to roughly 22,000 categories.

**Architecture:**

8 learned layers – 5 convolutional layers + 3 fully-connected layers



**Architecture of AlexNet**

The **first** convolutional layer filters the 224\*224\*3 input image with 96 kernels of size 11\*11\*3 with a stride of 4 pixels.

The **second** convolutional layer takes as input the output of the first convolutional layer and filters it with 256 kernels of size 5\*5\*48.

The **third, fourth, and fifth** convolutional layers are connected to one another without any intervening pooling or normalization layers.

The **third** convolutional layer has 384 kernels of size 3\*3\*256 connected to the outputs of the second convolutional layer.

The **fourth** convolutional layer has 384 kernels of size 3\*3\*192.

The **fifth** convolutional layer has 256 kernels of size 3\*3\*192.

The **fully-connected layers** have 4096 neurons each.

**Feature:**

1. **Rectified Linear Units (ReLU) Nonlinearity**

 shorter training time with gradient descent

Reference: [Deep Sparse Rectifier Neural Networks](http://proceedings.mlr.press/v15/glorot11a.html)

1. **Training on Multiple GPUs**

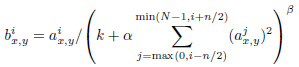
Use 2 GPUs and put half of the kernels (or neurons) on each GPU. The GPUs communicate only in certain layers. The Two-GPU net takes less time to train.

Kernels on GPU 1 are largely color-agnostic.

Kernels on GPU 2 are largely color-specific.

1. Brightness normalization - Local Response Normalization

* **Aids generalization**



1. **Overlapping Pooling**

a pooling layer consists of a grid of pooling units spaced s pixels apart, each summarizing a neighborhood of size z\*z centered at the location of the pooling unit.

* s<z

1. **Reducing Overfitting**

* Enlarge the dataset using label-preserving transformations

1. generate image translations and horizontal reflections

[question] We do this by extracting random 224\*224 patches from the 256\*256 images and training our network on these extracted patches4. This increases the size of our training set by a factor of 2048.

2\*(256-224)^2=2048 but why?

1. alter the intensities of the RGB channels in training images

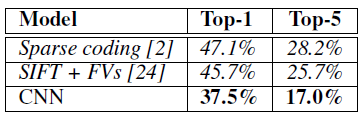
perform Principal component analysis (PCA) on the set of RGB pixel values

1. **Dropout**

set to zero the output of each hidden neuron with probability 0.5

[question] choose only half of the neurons at each layer?

Reference: [Improving neural networks by preventing co-adaptation of feature detectors](https://arxiv.org/pdf/1207.0580.pdf)



1. **VGG (2014)** trained 6 VGG-networks with comparison

Original paper: [Very Deep Convolutional Networks for Large-Scale Image Recognition](https://arxiv.org/pdf/1409.1556.pdf)

**Architecture:**

A stack of convolutional layers - 3 Fully-Connected (FC) layers - soft-max layer

Input - a fixed-size 224 × 224 RGB image

**FC layers**: The **first two** have 4096 channels each, the **third** performs 1000-way ILSVRC classification and thus contains 1000 channels. FC layers are the same from A-E.

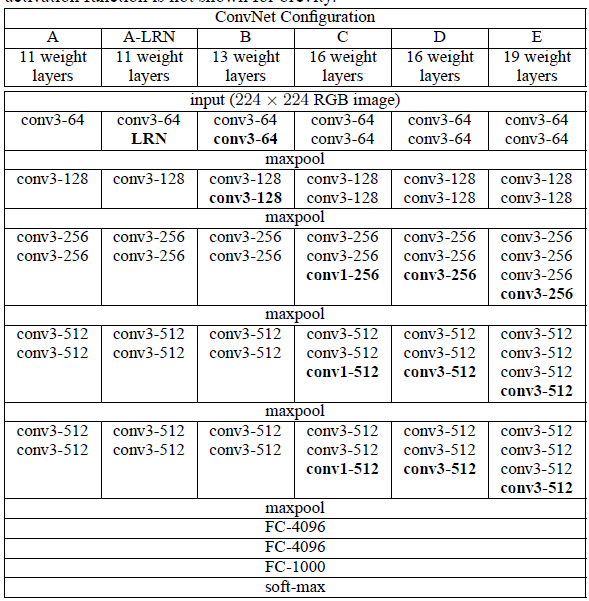
All hidden layers are equipped with the rectification (ReLU) non-linearity.

None of our networks (except for VGG-A) contain Local Response Normalisation (LRN) normalization.

**Features:**

Difference: use small 3\*3 receptive fields throughout the whole net instead of relatively large receptive fields in the first conv. layers.

1. incorporate three non-linear rectification layers instead of a single one
2. decrease the number of parameters



**ConvNet configurations (VGGs)**

1. **GoogLeNet/Inception (2014)**

Original paper: [Going deeper with convolutions](https://arxiv.org/pdf/1409.4842v1.pdf)

Idea derives from “Network in network”.

Talked about the trend at that moment: increase the number of layers and layer size, use dropout to address the problem of overfitting

Main idea to improve the performance of deep neural networks: **increasing the size**

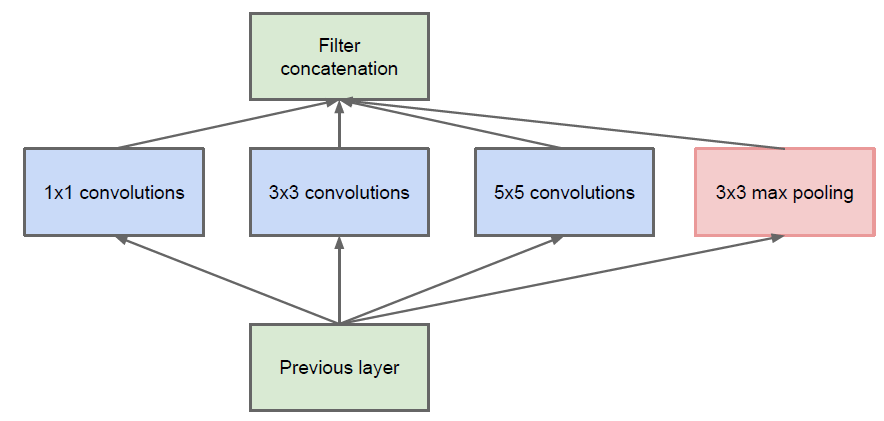
* **2 Drawbacks**

1. a larger number of parameters so the network more prone to overfitting
2. dramatically increased use of computational resources

* solution: ultimately moving from fully connected to sparsely connected architectures

**Architecture of Inception:**

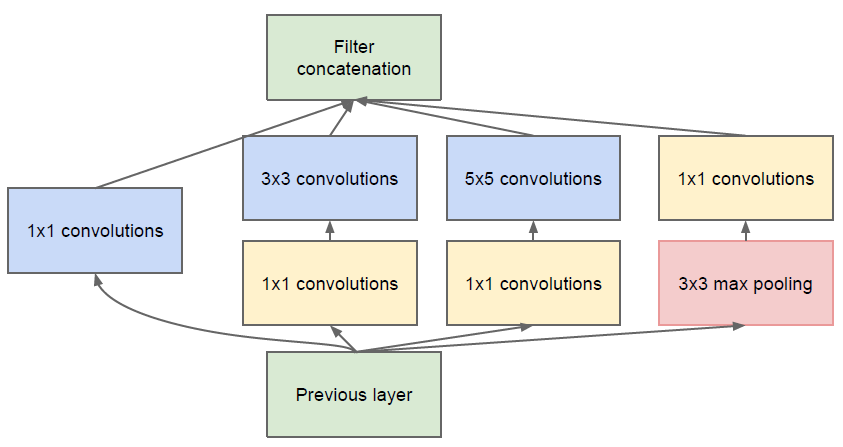
Main idea of the Inception architecture: finding out how an optimal local sparse structure in a convolutional vision network can be approximated and covered by readily available dense components.



**Inception module, naive version**

Problem: computational blow up because of large number of parameters

* applying dimension reductions and projections



**Inception module with dimension reductions**

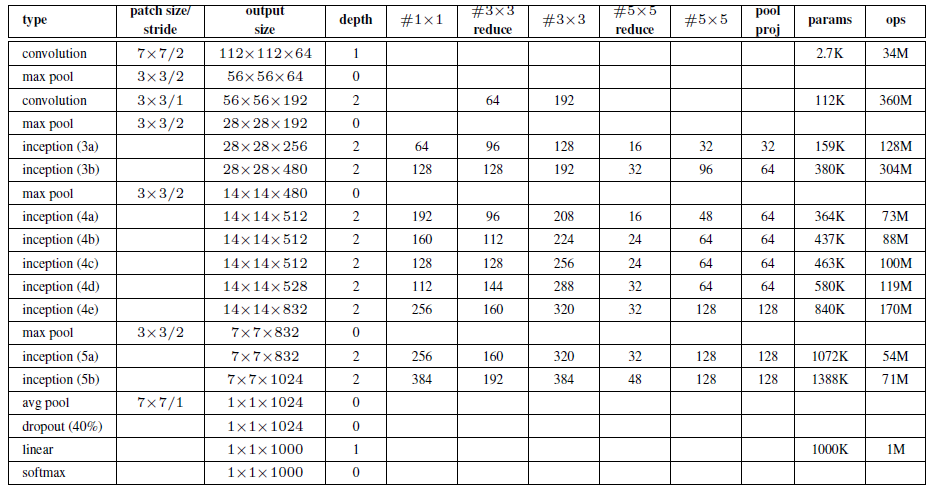
**Inception network** is a network consisting of modules of the above type stacked upon each other, with occasional max-pooling layers with stride 2 to halve the resolution of the grid.

**Architecture of GoogLeNet:**

22 layers deep

The overall number of layers used is about 100 - depends on the machine learning infrastructure system used

All the convolutions use ReLU. The size of the receptive field is 224\*224 taking RGB color channels with mean subtraction.



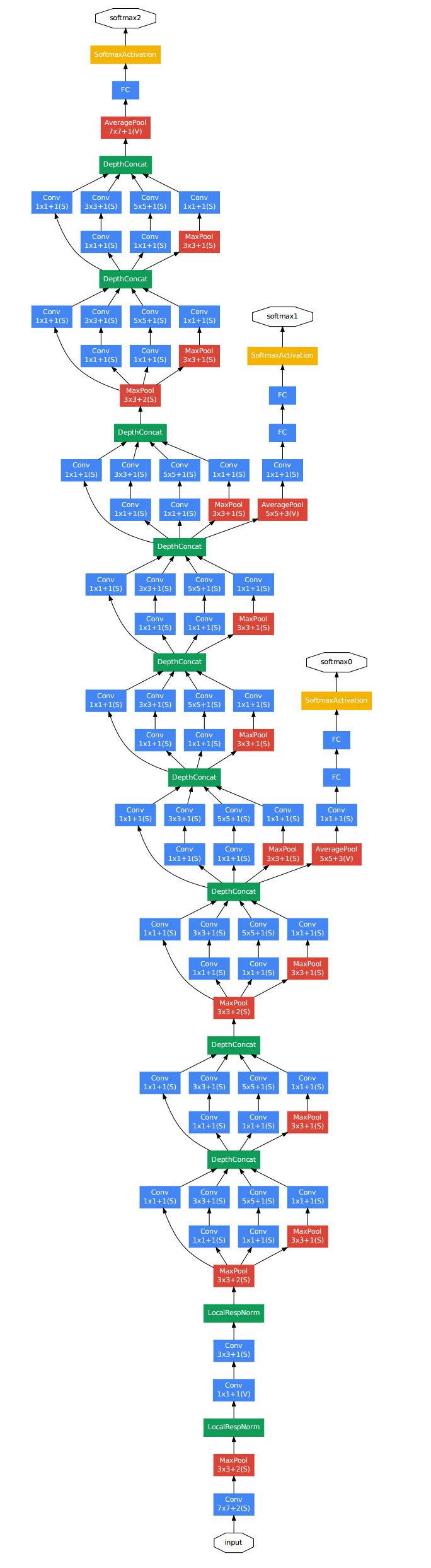
**GoogLeNet incarnation of the Inception architecture**

**Feature:**

add auxiliary classifiers connected to these intermediate layers

* An average pooling layer with 5\*5 filter size and stride 3, resulting in an 4\*4\*512 output for the (4a), and 4\*4\*528 for the (4d) stage.
* A 1\*1 convolution with 128 filters for dimension reduction and rectified linear activation.
* A fully connected layer with 1024 units and rectified linear activation.
* A dropout layer with 70% ratio of dropped outputs.
* A linear layer with softmax loss as the classifier (predicting

Another paper about GoogLeNet: [Rethinking the Inception Architecture for Computer Vision](https://arxiv.org/pdf/1512.00567.pdf) (Inception v2)



1. **ResNet (2015)** introducing a residual learning framework (to ease the training)

Original paper: [Deep Residual Learning for Image Recognition](https://arxiv.org/pdf/1512.03385.pdf)

**Degradation** problem: with the network depth increasing, accuracy gets saturated and then degrades rapidly.

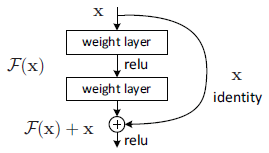
Idea: **Identity Mapping by Shortcuts**

 - the desired underlying mapping; - the residual

Let the stacked nonlinear layers fit another mapping 

To optimize the residual mapping is easier than to optimize the original

* realized by feedforward neural networks with “shortcut connections”



**Structure of Residual learning**

ResNet are easy to optimize, but the counterpart “plain” nets exhibit higher training error when the depth increases.

ResNet can easily enjoy accuracy gains from greatly increased depth, producing results substantially better than previous networks.

**Architecture:**

1. **Plain Network** - inspired by VGG (fewer filters and lower complexity than VGG nets)

convolutional layers - global average pooling layer - fully-connected layer with softmax

The convolutional layers mostly have 3\*3 filters.

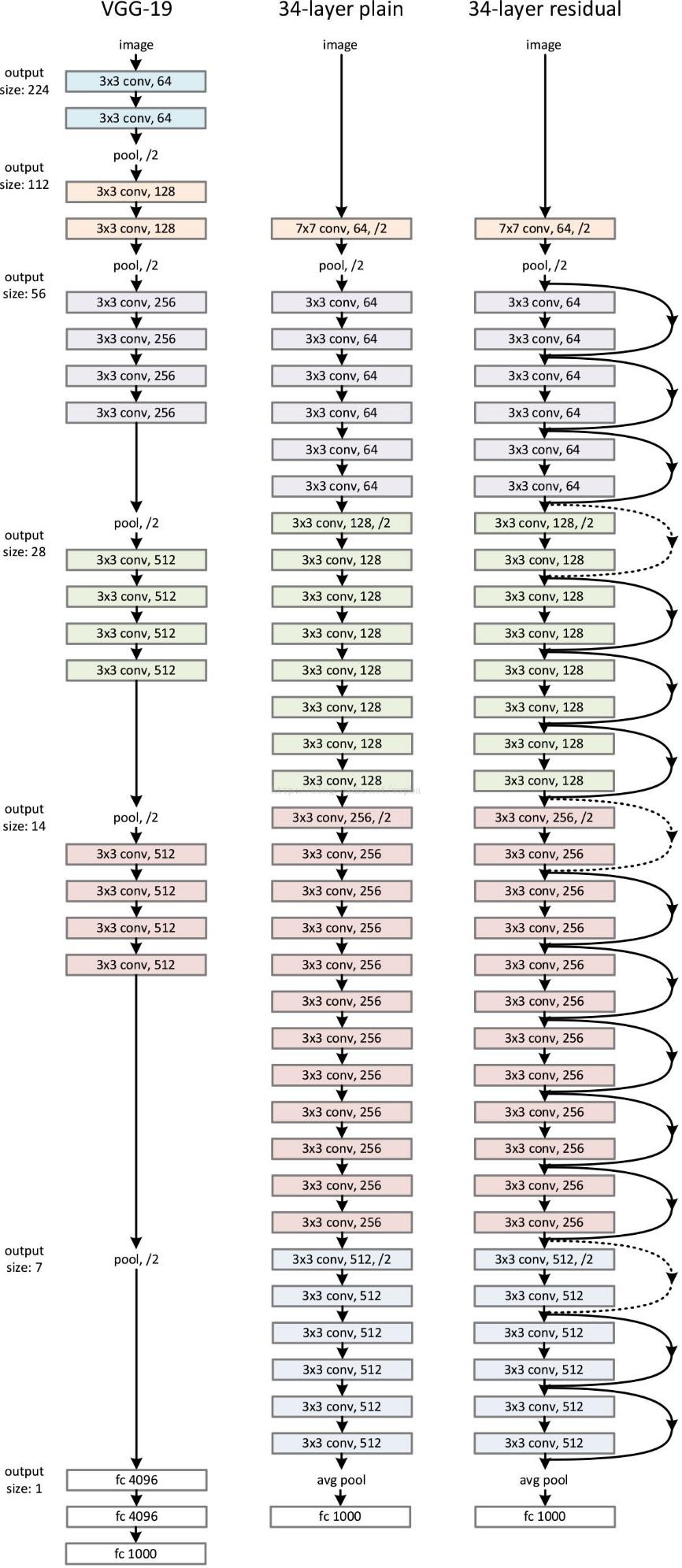
The total number of weighted layers – 34

2 rules for the convolutional layers

1. for the same output feature map size, the layers have the same number of filters.
2. if the feature map size is halved, the number of filters is doubled so as to preserve the time complexity per layer.
3. **Residual Network** (insert shortcut connections)

2 options:

1. The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions.
2. The projection shortcut in  is used to match dimensions



**Architecture of ResNet**

1. **SqueezeNet (2016)**

Original paper: [SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size](https://arxiv.org/pdf/1602.07360.pdf)

**3 strategies:**

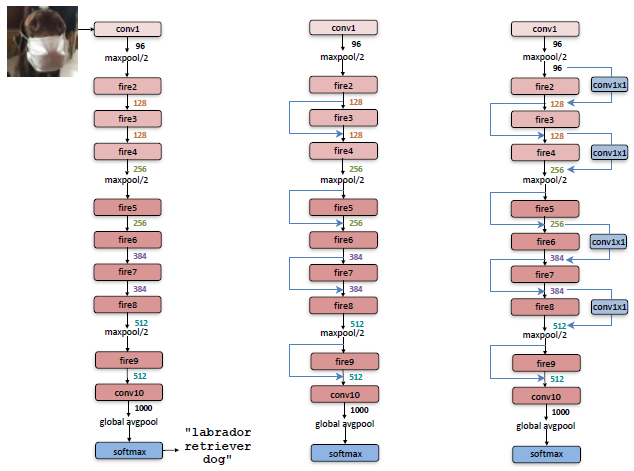
1. Replace 3x3 filters with 1x1 filters – fewer parameters
2. Decrease the number of input channels to 3x3 filters - fewer parameters
3. Downsample late in the network so that convolution layers have large activation maps – higher classification accuracy

**The Fire module** - a squeeze convolution layer (1x1 filters), feeding into an expand layer that has a mix of 1x1 and 3x3 convolution filters;

**Architecture:**

standalone convolution layer (conv1) - 8 Fire modules (fire2-9) - final conv layer (conv10).

SqueezeNet performs max-pooling with a stride of 2 after layers conv1, fire4, fire8, and conv10.



**Macroarchitectural view of SqueezeNet architecture**

**SqueezeNet; SqueezeNet with simple bypass; SqueezeNet with complex bypass**

**Features:**

* added a 1-pixel border of zero-padding in the input data to 3x3 filters of expand modules.
* ReLU applied
* Dropoutwith a ratio of 50% applied after the fire9 module.
* lack of fully-connected layers in SqueezeNet – inspired by Network in network

1. **Xception (2016)** Inception v3

Original paper: [Xception: Deep Learning with Depthwise Separable Convolutions](https://arxiv.org/pdf/1610.02357.pdf)

Core idea: the depth-wise separable convolutions

Can be seen as the pre-work for MobileNet

1. **MobileNet (2017)** depth-wise separable convolutions

Original paper: [MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications](https://arxiv.org/pdf/1704.04861.pdf)

**Depthwise separable convolutions** - a form of factorized convolutions which factorize a standard convolution into a depth-wise convolution and a 1\*1 convolution called a pointwise convolution. – applies a single filter to each input channel.

made up of two layers:

1. depthwise convolutions –apply a single filter per each input channel



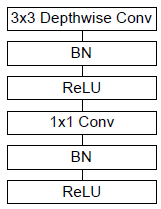
1. pointwise convolutions - create a linear combination of the output of the depthwise layer.

**Architecture:**

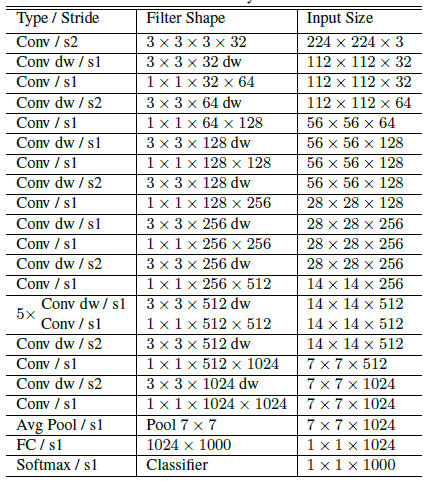
depthwise separable convolutions (first layer fully connected)

MobileNet has 28 layers.

use both batchnorm and ReLU nonlinearities for both layers.



**Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU**



**MobileNet Body Architecture**

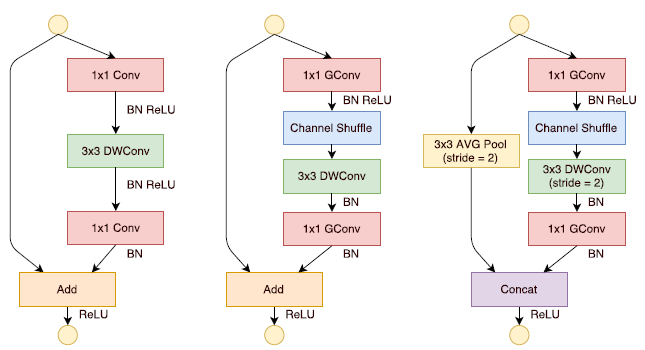
1. **ShuffleNet (2017)** pointwise group convolution + channel shuffle

Original paper: [ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices](https://arxiv.org/pdf/1707.01083.pdf)

2 operations: pointwise group convolution + channel shuffle

- reduce computation cost while maintaining accuracy.

**pointwise group convolution** : generalizes group convolution and depthwise separable convolution in a novel form

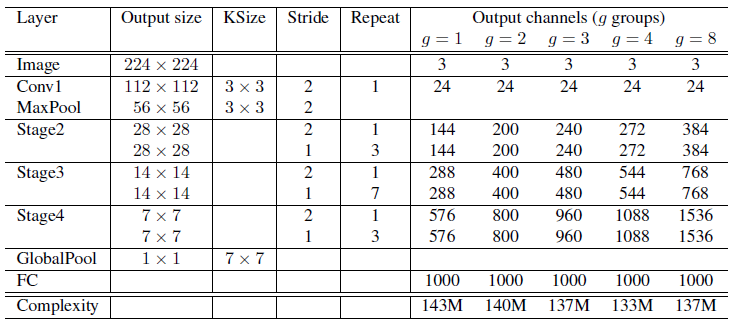


**ShuffleNet Units**

**bottleneck unit with depthwise convolution; ShuffleNet unit with pointwise group**

**convolution; ShuffleNet unit with stride = 2**

**Architecture:**



**ShuffleNet architecture**

**Further reading:**

ResNeXt (2016): [Aggregated residual transformations for deep neural networks](https://arxiv.org/pdf/1611.05431.pdf)

DenseNet (2016): [Densely Connected Convolutional Networks](https://arxiv.org/pdf/1608.06993.pdf)

**Datasets:**

1. [COCO](http://cocodataset.org/#home)

Common Objects in Context: a large-scale object detection, segmentation, and captioning dataset.

1. [ImageNet](http://www.image-net.org/)

Large Scale Visual Recognition Challenge ([ILSVRC](http://www.image-net.org/challenges/LSVRC/))

1. [PASCAL](http://host.robots.ox.ac.uk/pascal/VOC/)

The PASCAL Visual Object Classes dataset

1. [CIFAR 10/100](https://www.cs.toronto.edu/~kriz/cifar.html)
2. [NORB](https://cs.nyu.edu/~ylclab/data/norb-v1.0/)

Object Recognition Dataset

1. [MNIST](http://yann.lecun.com/exdb/mnist/) (handwritten digits) – very old