Analysis and Design of Deep Neural Networks

Chapter 3
Deep Architectures and
Layer Wise Analysis and Design Algorithms

Fall 2023

Ahmad Kalhor-University of Tehran

3. Deep Architectures and Layer Wise Analysis and Design Algorithms

3.1 Architecture of CNNs

3.2 Layer Wise Analysis Algorithms

- Layer-wise Model evaluation
- Pre-train Model ranking
- Model Confidence and Guarantee

3.3 Architecture of Region Based CNNs

3.4 Layer Wise Design Algorithms_part1

- Model Compressing

3.5 Architecture of Transformers

3.6 Layer Wise Design Algorithms_part2

- Layer-wise forward learning (supervised, self supervised, unsupervised)
- Layer-wise branching/Fusion

3.7 Other Layer-wise Algorithms in Literature

3.1 Architectures of CNNs

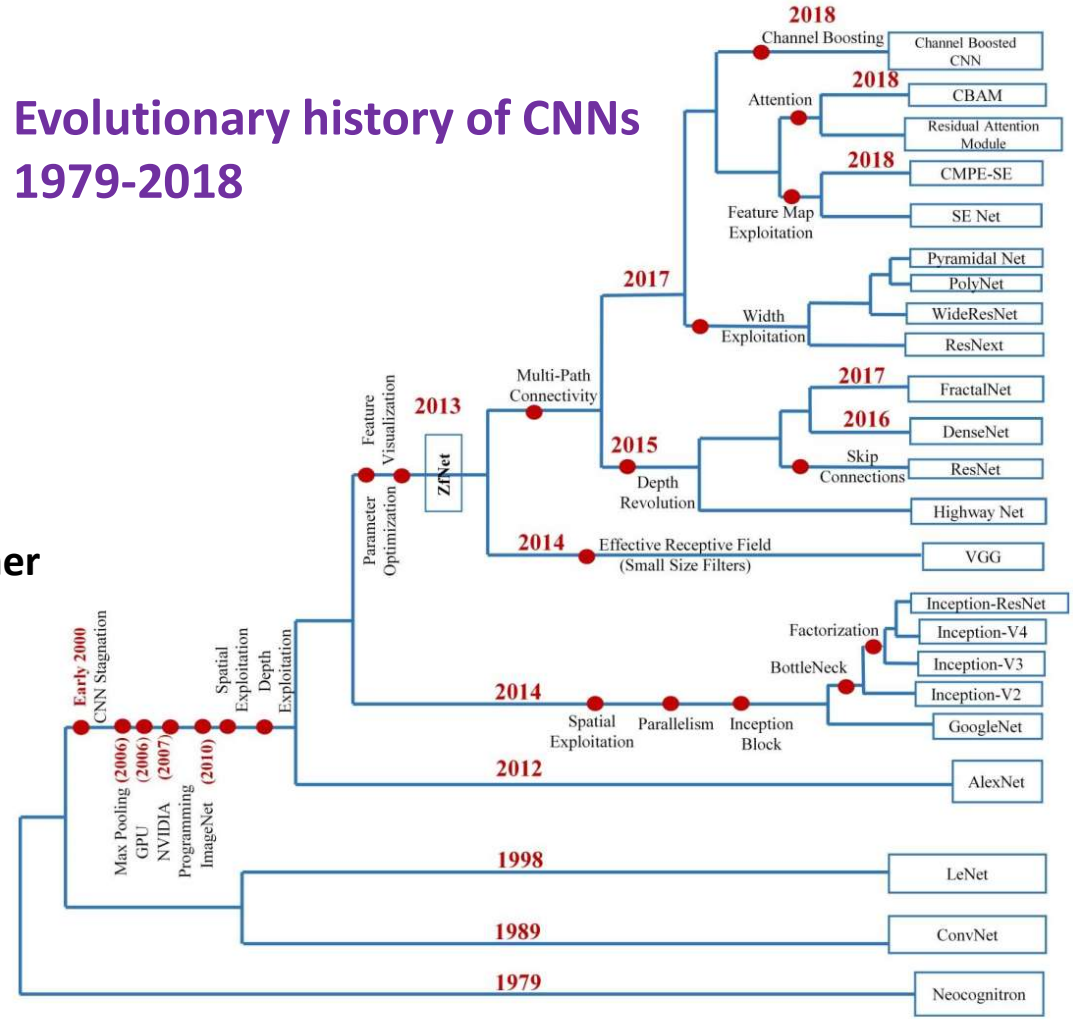
3.1.1 CNNs

Neural Networks which use convolution layers to filter and extract features from signals to solve a classification or regression problem.

In 1989, LeCuN et al. proposed the first multilayered CNN named ConvNet, whose origin rooted in Fukushima's Neocognitron (Fukushima and Miyake 1982; Fukushima 1988). **Japanese handwritten character recognition and other pattern recognition tasks**

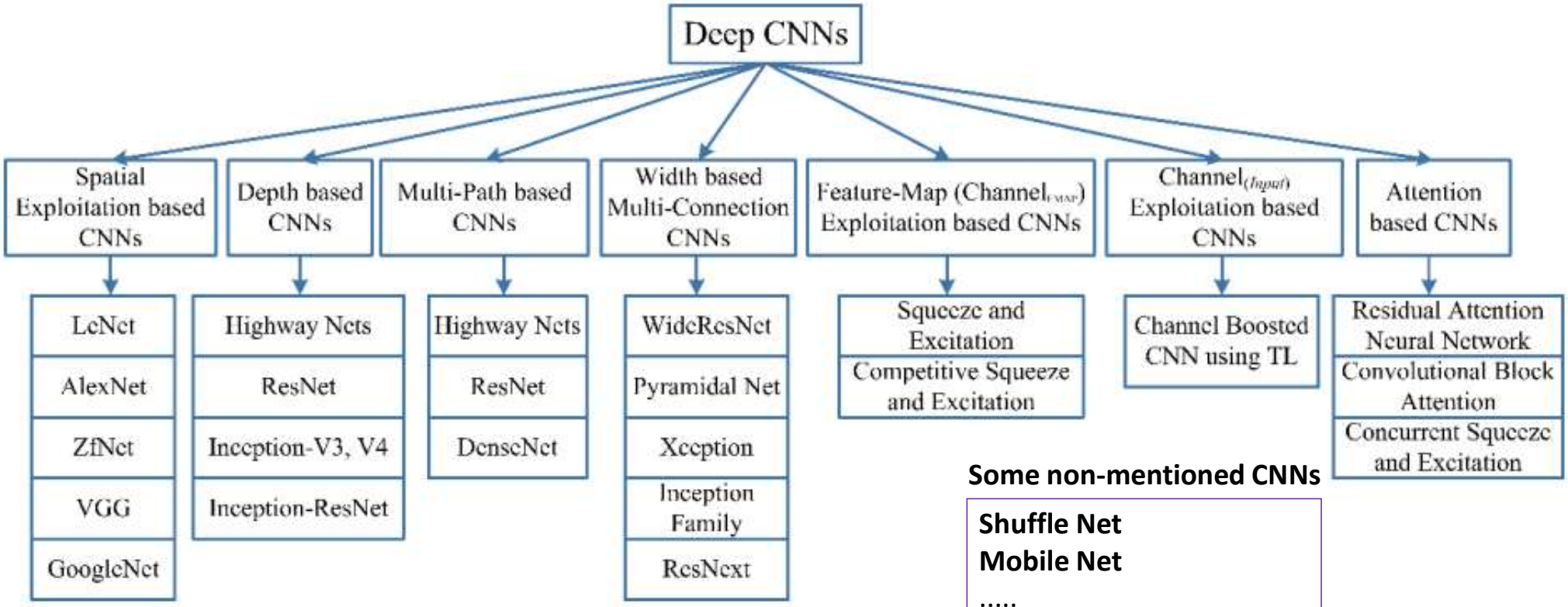
In 1998, LeCuN proposed an improved version of ConvNet, which was famously known as LeNet-5, and it started the use of CNN in classifying characters in a document recognition related applications (LeCun et al. 1995, 1998).

Evolutionary history of CNNs 1979-2018



*Assifullah khan, et al. 2020

Taxonomy of deep CNN architectures showing seven different categories



Some non-mentioned CNNs

Shuffle Net
Mobile Net
.....
Efficient Net
Transformer-CNN

(1) Spatial Exploitation based CNNs

CNNs which improve themselves by exploring different levels of neighborhood (spatial) correlation among pixels of an input image or units of feature maps by employing different filter sizes

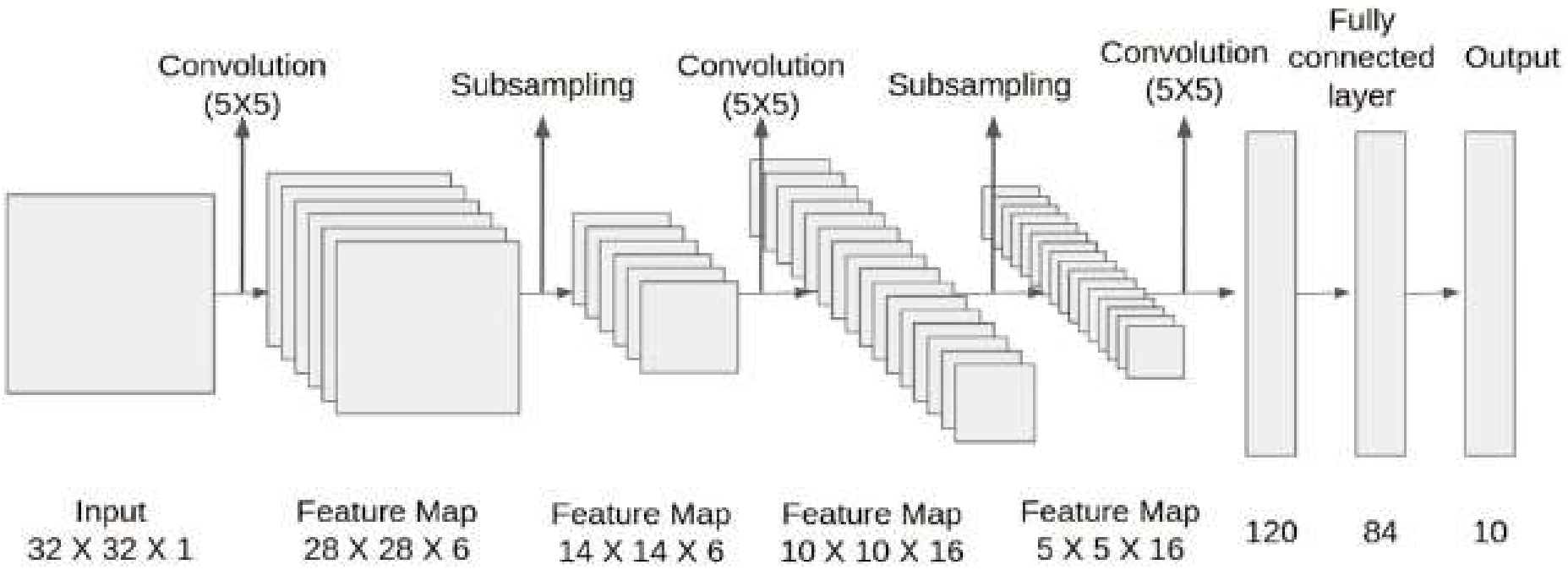
CNNs: **LeNet** **AlexNet** **ZFNet** **VGG** **GoogleNet**

.

Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
LeNet	1998	- First popular CNN architecture	0.060 M	[dist]MNIST: 0.8 MNIST: 0.95	5	Spatial Exploitation	(LeCun et al. 1995)
AlexNet	2012	- Deeper and wider than the LeNet - Uses Relu, dropout and overlap Pooling - GPUs NVIDIA GTX 580	60 M	ImageNet: 16.4	8	Spatial Exploitation	(Krizhevsky et al. 2012)
ZfNet	2014	-Visualization of intermediate layers	60 M	ImageNet: 11.7	8	Spatial Exploitation	(Zeiler and Fergus 2013)
VGG	2014	- Homogenous topology - Uses small size kernels	138 M	ImageNet: 7.3	19	Spatial Exploitation	(Simonyan and Zisserman 2015)
GoogLeNet	2015	- Introduced block concept - Split transform and merge idea	4 M	ImageNet: 6.7	22	Spatial Exploitation	(Szegedy et al. 2015)

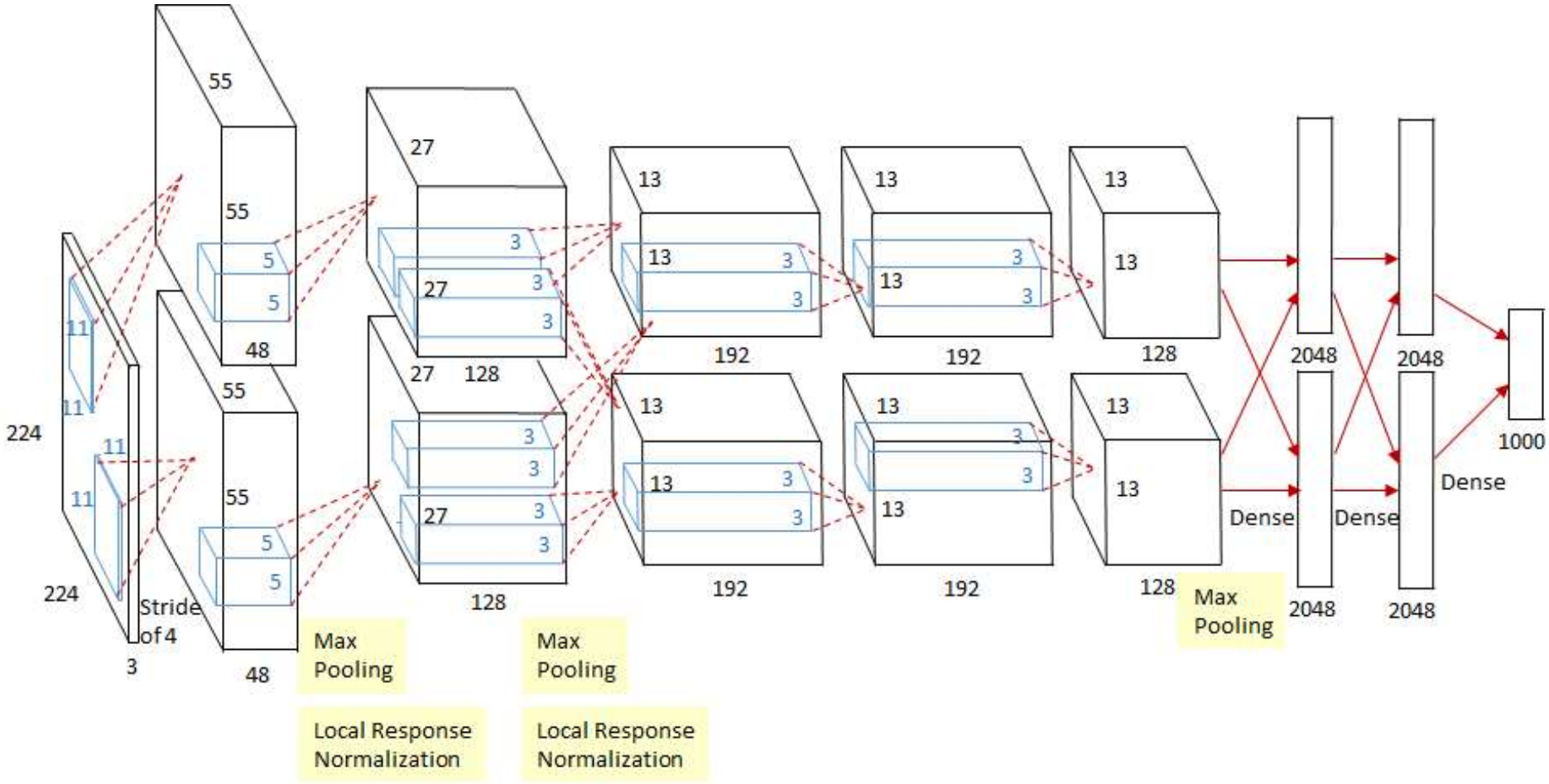
Lenet-5

First popular CNN architecture



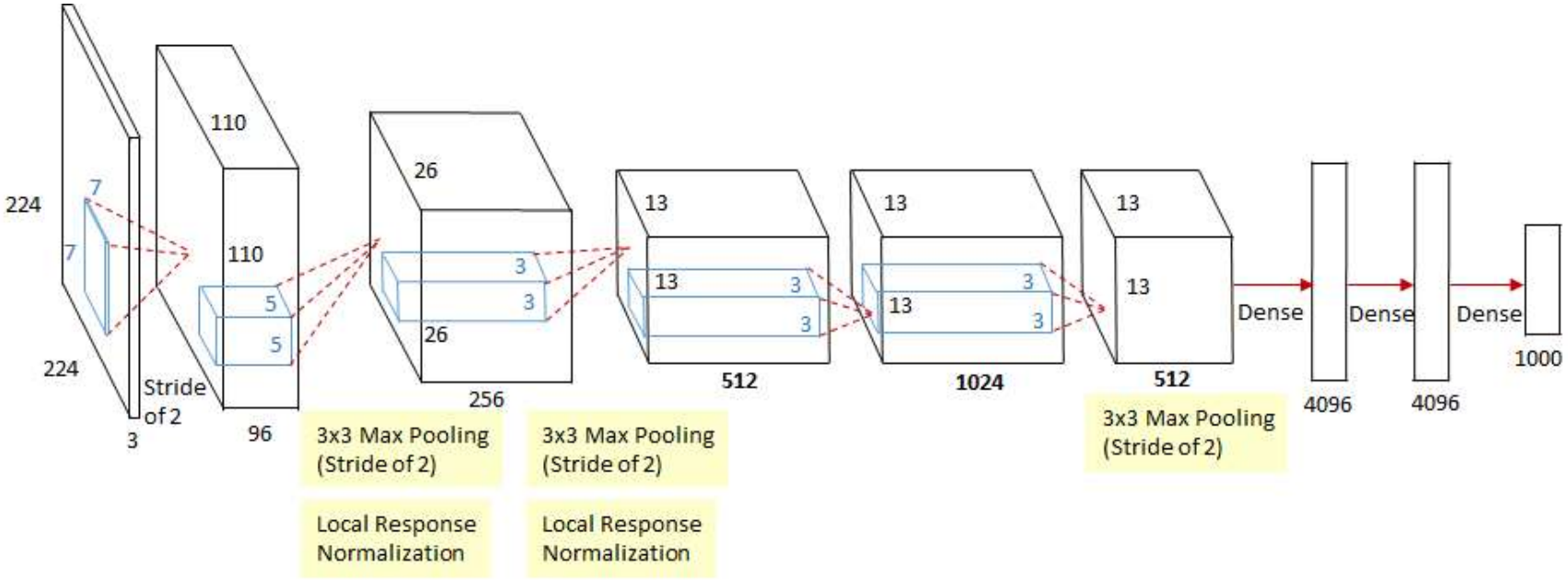
AlexNet

- Deeper and wider than the LeNet
- Uses Relu, dropout and overlap Pooling



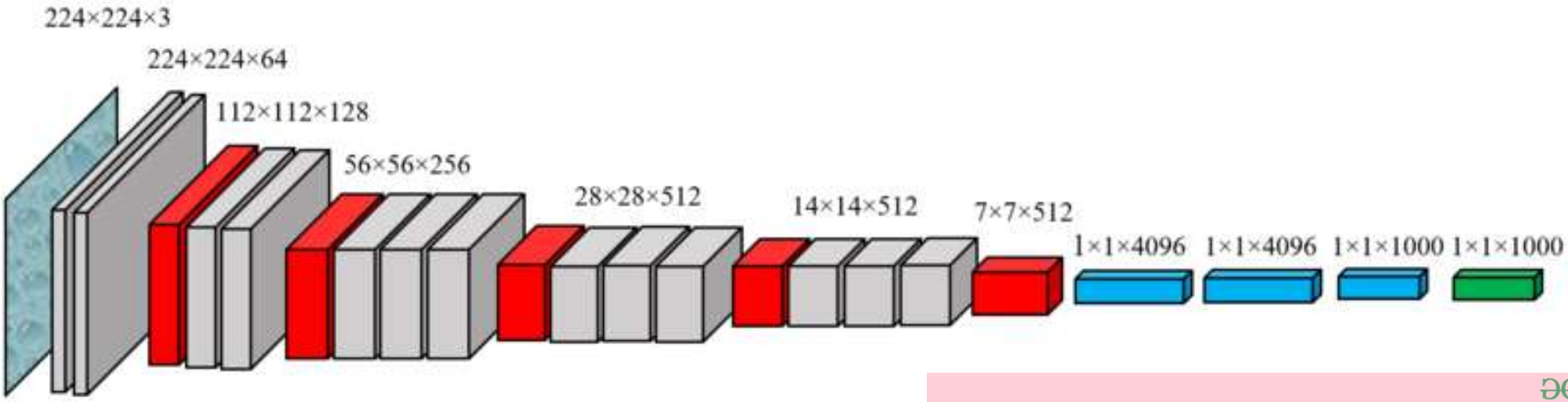
ZFNET

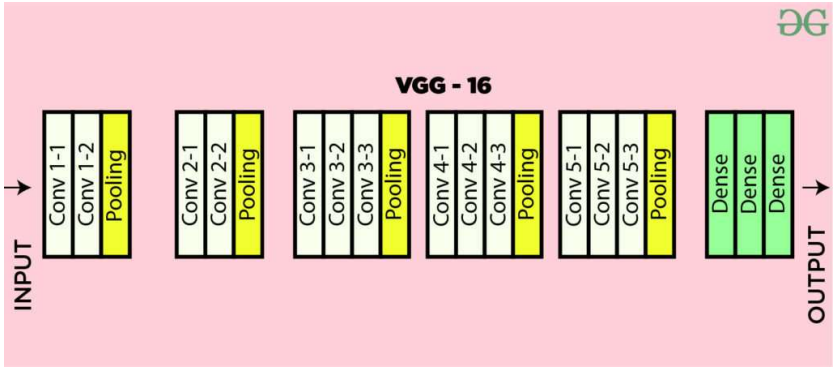
Visualization of intermediate layers



VGG

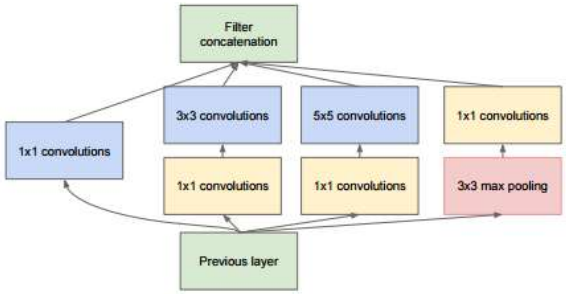
- Homogenous topology
- Uses small size kernels





GoogLeNet

- Introduced block concept
- Split transform and merge idea



(b) Inception module with dimension reductions

Auxiliary Classifiers are type of architectural component that seek to improve the convergence of very deep networks.
They are classifier heads we attach to layers before the end of the network.
The motivation is to push useful gradients to the lower layers to make them immediately useful and improve the convergence during training by combatting the vanishing gradient problem.
They are notably used in the Inception family of convolutional neural networks.

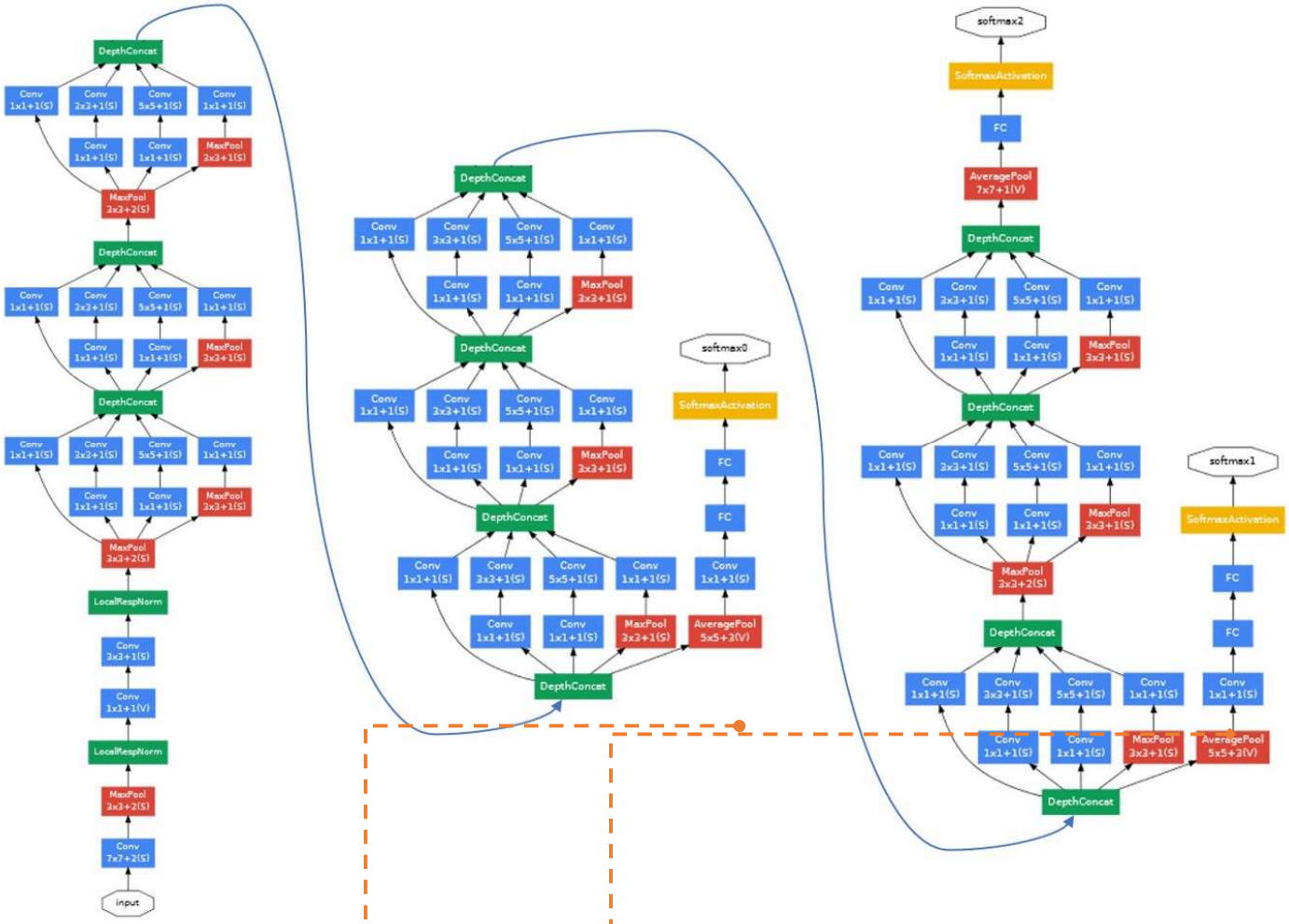


Table 5a Major challenges associated with implementation of Spatial exploitation based CNN architectures.

Spatial Exploitation	As convolutional operation considers the neighborhood (correlation) of input pixels, therefore different levels of correlation can be explored by using different filter sizes.	
Architecture	**Strength**	**Gaps**
LeNet	• Exploited spatial correlation to reduce the computation and number of parameters • Automatic learning of feature hierarchies	• Poor scaling to diverse classes of images • Large size filters • Low level feature extraction
AlexNet	• Low, mid and high-level feature extraction using large and small size filters on initial (5x5 and 11x11) and last layers (3x3) • Give an idea of deep and wide CNN architecture • Introduced regularization in CNN • Started parallel use of GPUs as an accelerator to deal with complex architectures	• Inactive neurons in the first and second layers • Aliasing artifacts in the learned feature-maps due to large filter size
ZfNet	• Introduced the idea of parameter tuning by visualizing the output of intermediate layers • Reduced both the filter size and stride in the first two layers of AlexNet	• Extra information processing is required for visualization
VGG	• Proposed an idea of effective receptive field • Gave the idea of simple and homogenous topology	• Use of computationally expensive fully connected layers
GoogLeNet	• Introduced the idea of using Mutiscale Filters within the layers • Gave a new idea of split, transform, and merge • Reduce the number of parameters by using bottleneck layer, global average-pooling at last layer and Sparse Connections • Use of auxiliary classifiers to improve the convergence rate	• Tedious parameter customization due to heterogeneous topology • May lose the useful information due to representational bottleneck

(2) Depth based CNNs

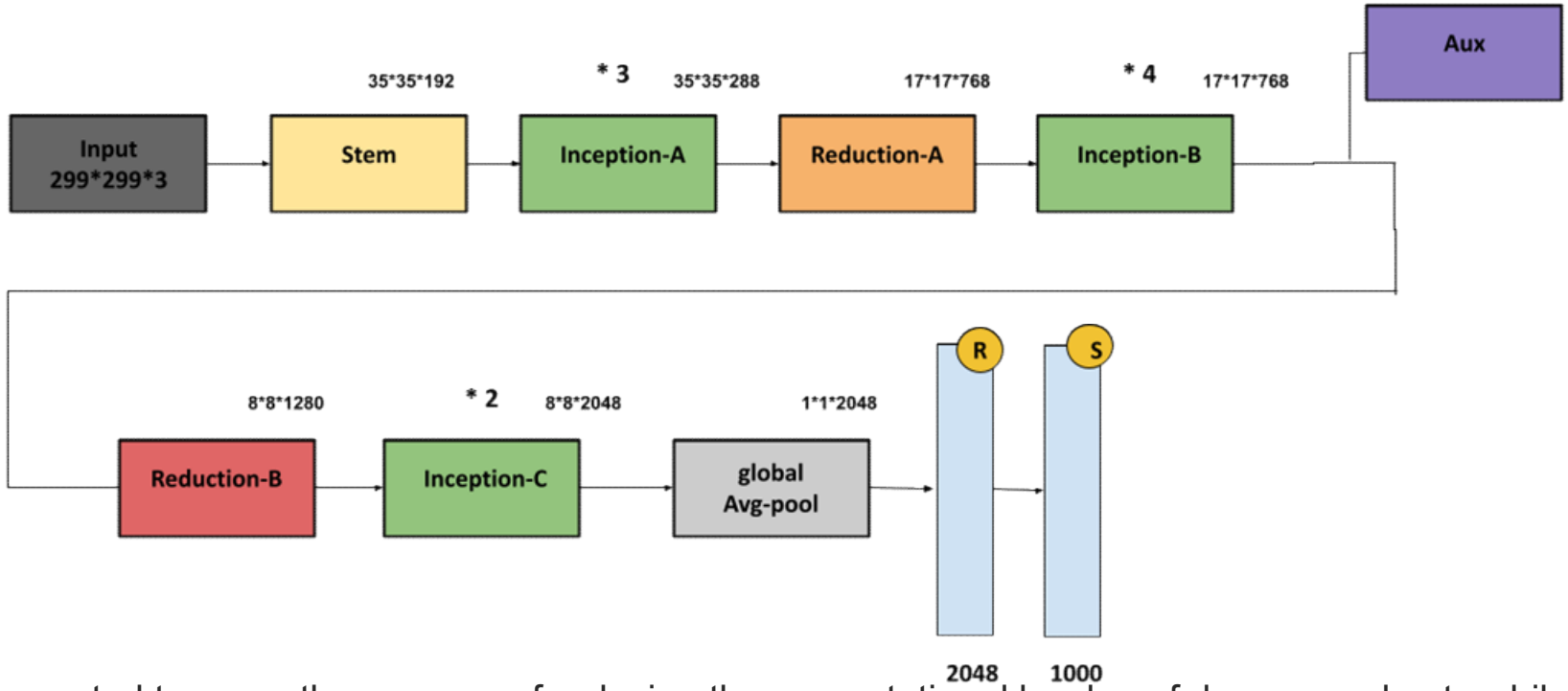
CNNs which improve themselves by increasing the depth to utilize more cascaded filters and to achieve better feature representation

CNNs: Inception-V3, V4 Inception-ResNet ResNet Highway Networks

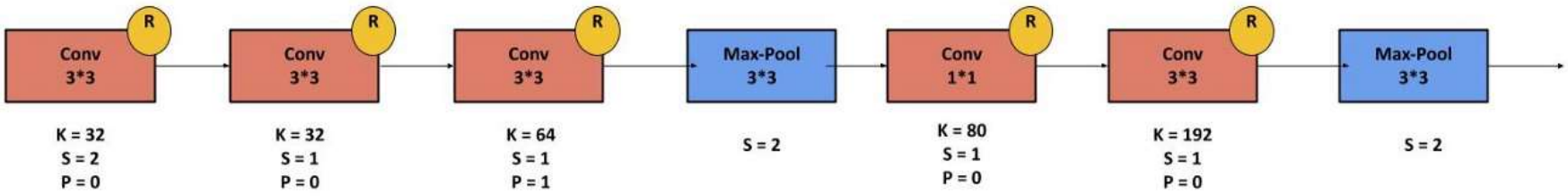
Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
Inception-V3	2015	- Handles the problem of a representational bottleneck - Replace large size filters with small filters	23.6 M	ImageNet: 3.5 Multi-Crop: 3.58 Single-Crop: 5.6	159	Depth + Width	(Szegedy et al. 2016b)
Highway Networks	2015	- Introduced an idea of Multi-path	2.3 M	CIFAR-10: 7.76	19	Depth + Multi-Path	(Srivastava et al. 2015a)
Inception-V4	2016	- Split transform and merge idea Uses asymmetric filters	35 M	ImageNet: 4.01	70	Depth +Width	(Szegedy et al. 2016a)
Inception-ResNet	2016	- Uses split transform merge idea and residual links	55.8M	ImageNet: 3.52	572	Depth + Width + Multi-Path	(Szegedy et al. 2016a)
ResNet	2016	- Residual learning - Identity mapping based skip connections	25.6 M 1.7 M	ImageNet: 3.6 CIFAR-10: 6.43	152 110	Depth + Multi-Path	(He et al. 2015a)

Inception V3

- Handles the problem of a representational bottleneck
-Using inception modules, Inceptions decease the representation size and avoid information missing
- Replace large size filters with small filters

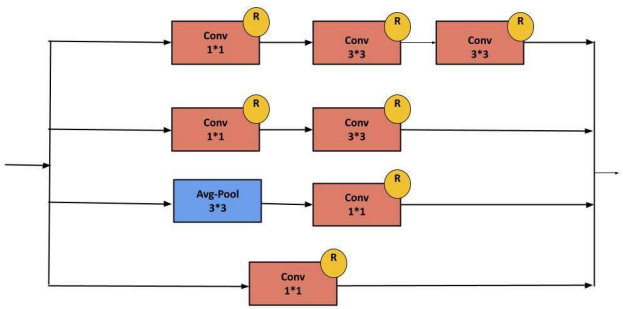


Inception is created to serve the purpose of reducing the computational burden of deep neural nets while obtaining state-of-art performance.

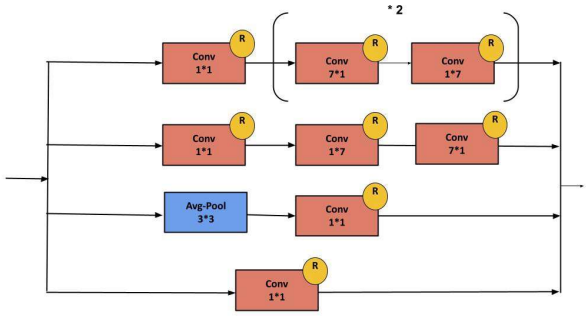


STEM

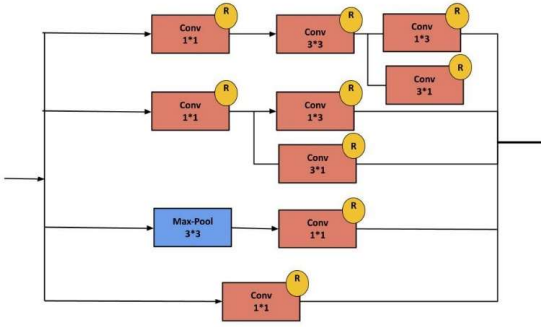
The Stem is a particular convolutional network module before the Inception-resnet blocks



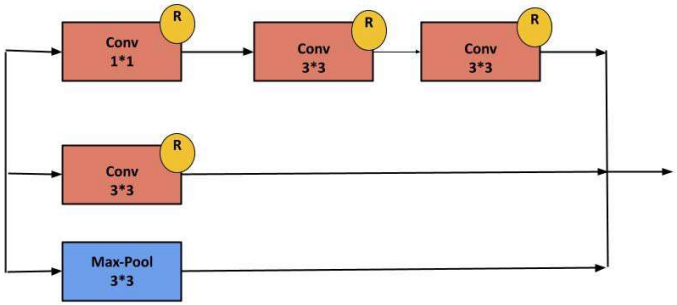
Inception-A Block :



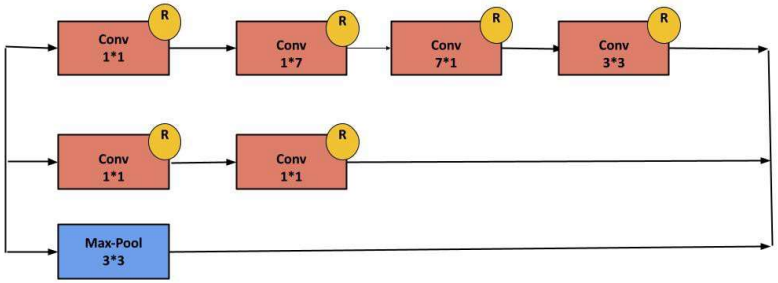
Inception-B Block :



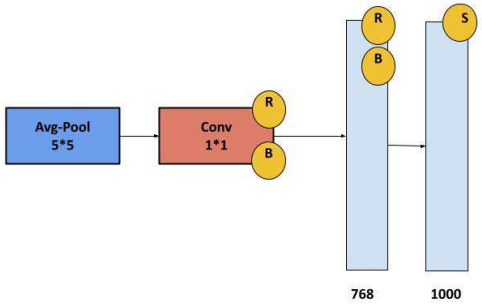
Inception-C Block :



Reduction-A Block :



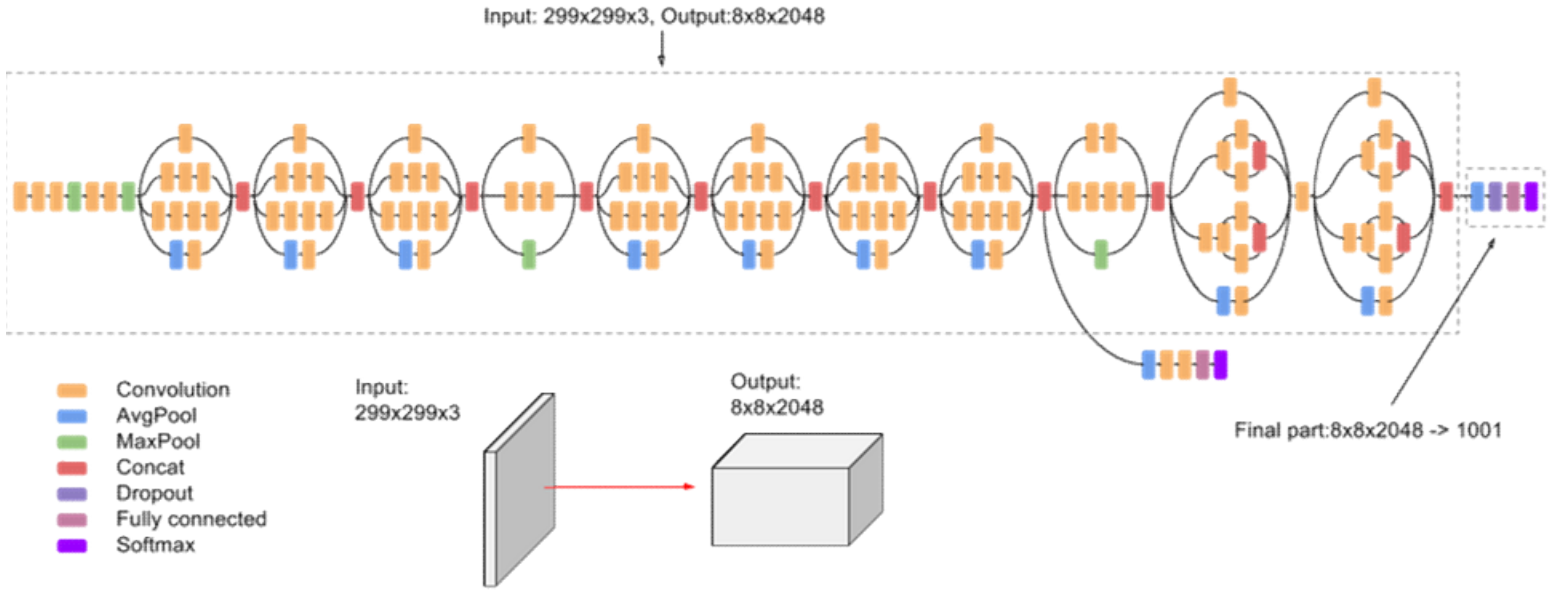
Reduction-B Block :



Auxiliary Classifier Block :

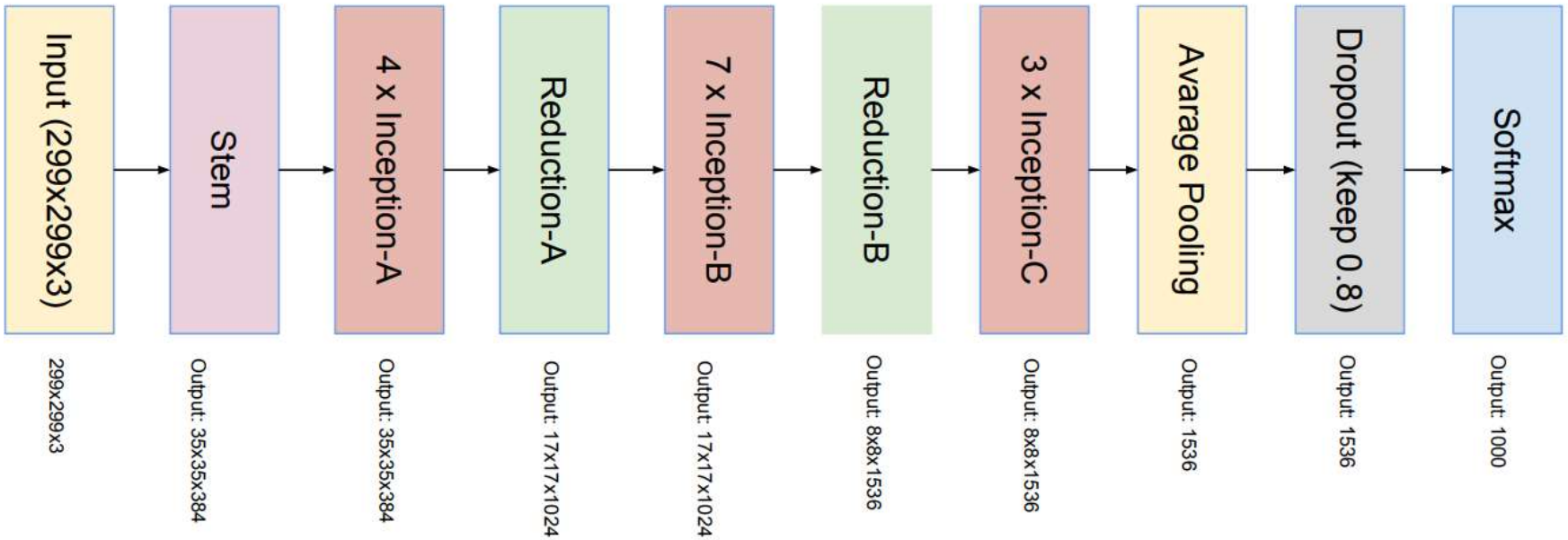
Auxiliary Classifiers are **type of architectural component that seek to improve the convergence of very deep networks**. They are classifier heads we attach to layers before the end of the network.

Inception 3

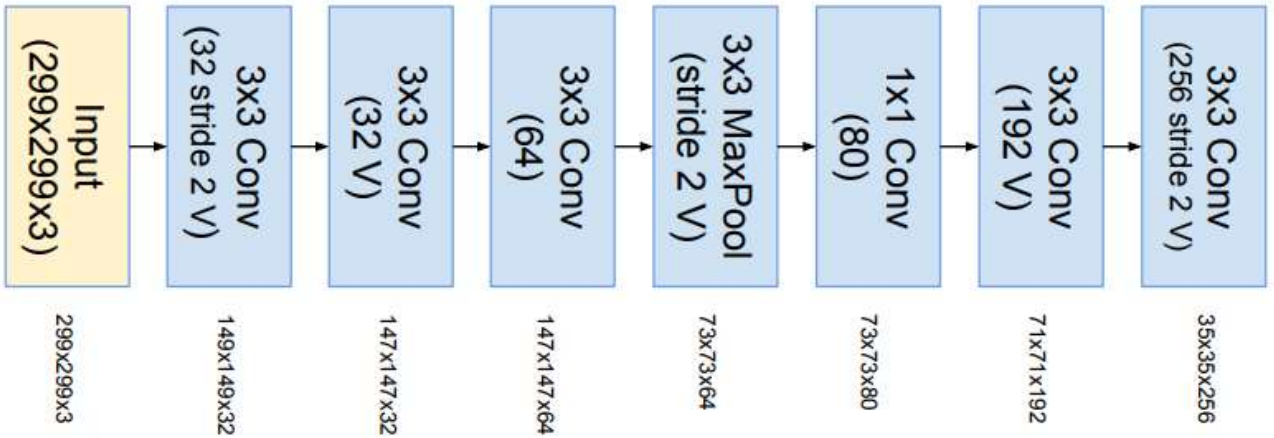

Input: 299x299x3, Output:8x8x2048
Convolution
AvgPool
MaxPool
Concat
Dropout
Fully connected
Softmax
Input:
299x299x3
Output:
8x8x2048
Final part:8x8x2048 -> 1001

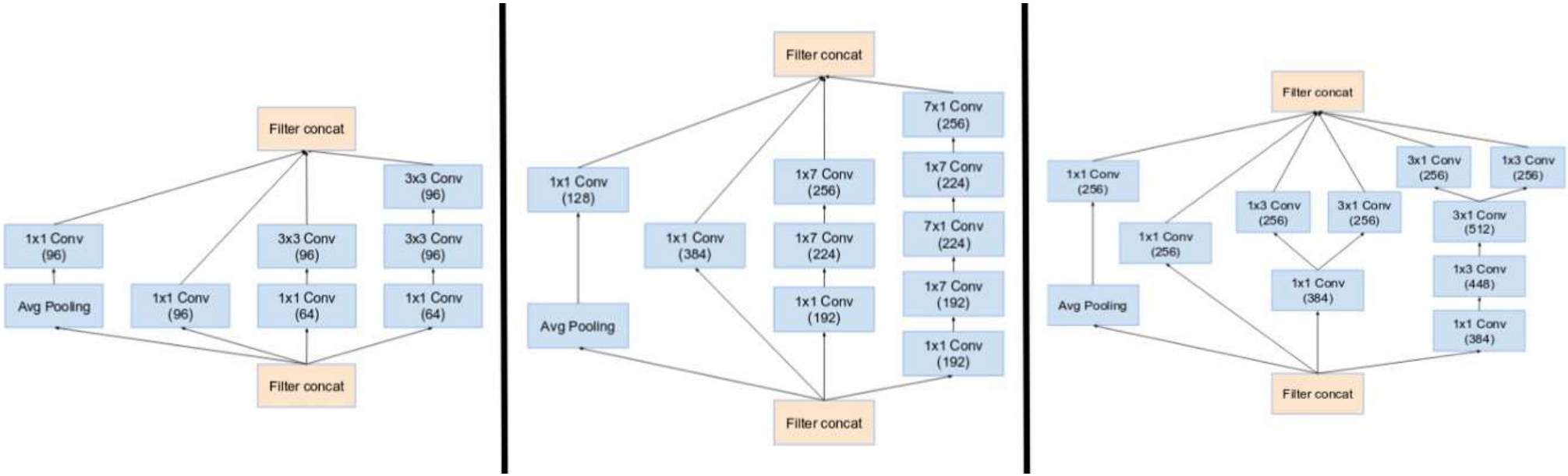
Inception 4

- Deep hierarchies features, Multi level feature representation
 -inception-v4 use more inception modules than Inception-v3.
- Split transform and merge idea Uses asymmetric filters

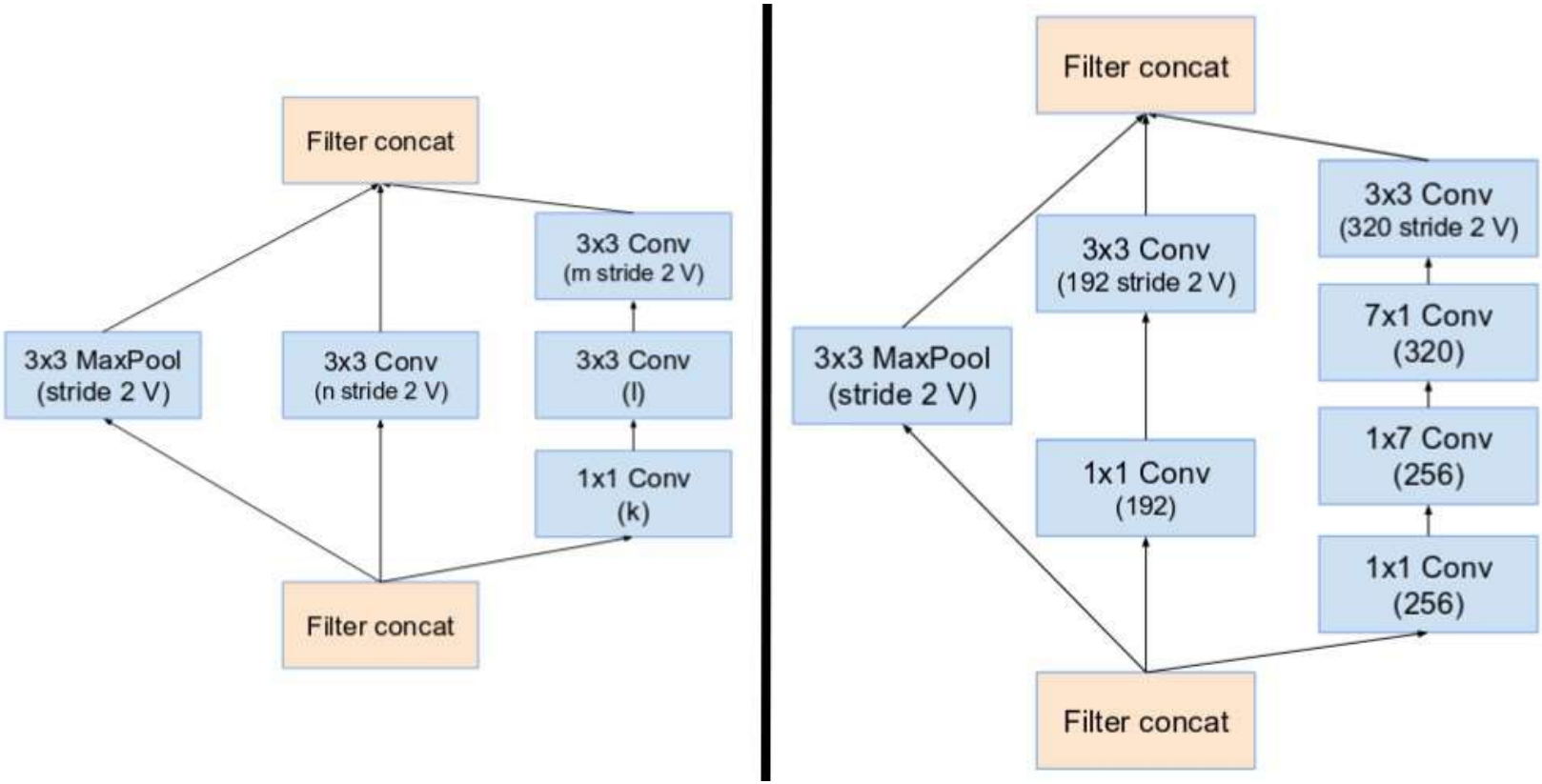


STEM


Input
(299x299x3)
299x299x3
3x3 Conv
(32 stride 2 V)
149x149x32
3x3 Conv
(32 V)
147x147x32
3x3 Conv
(64)
147x147x64
3x3 MaxPool
(stride 2 V)
73x73x64
1x1 Conv
(80)
73x73x80
3x3 Conv
(192 V)
71x71x192
3x3 Conv
(256 stride 2 V)
35x35x256



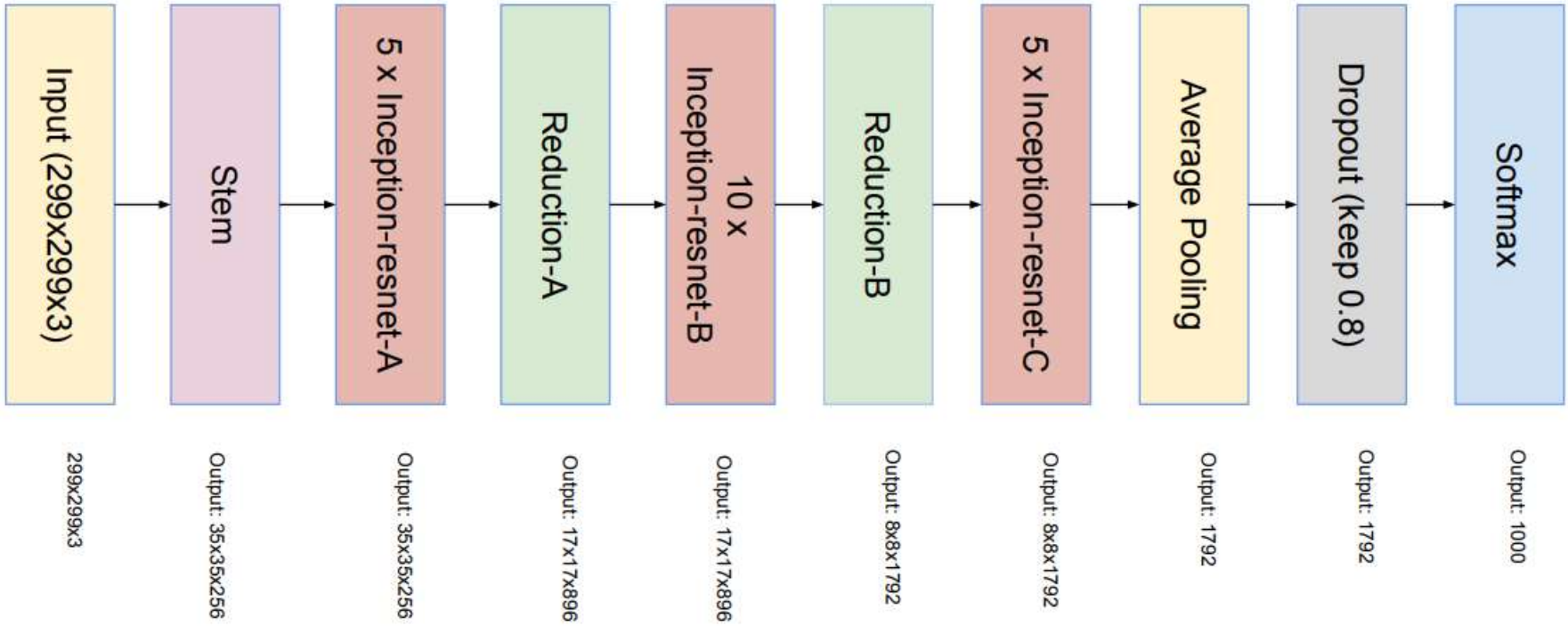
Inception Modules A, B, C of Inception-v4



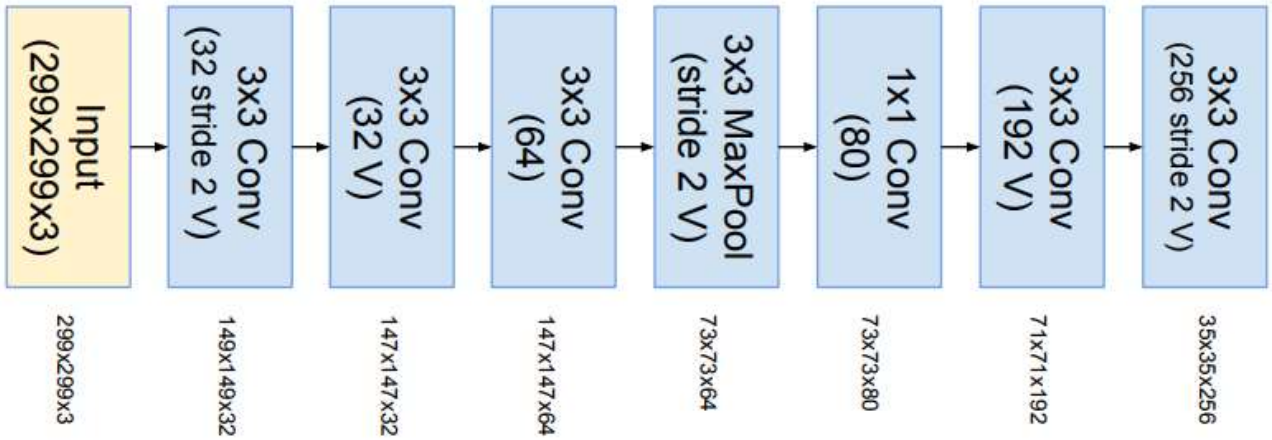
Reduction Blocks A, B of Inception-v4

Inception Resnet

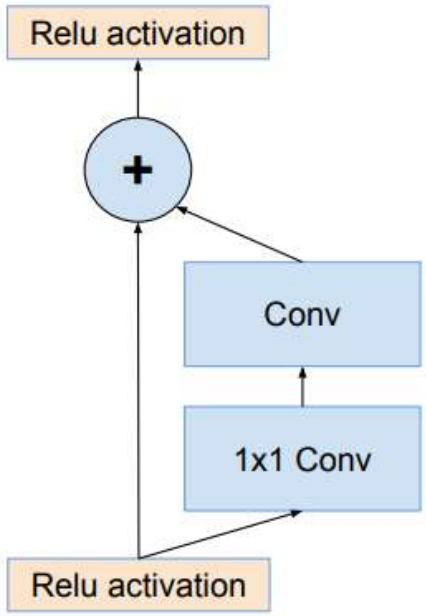
- Uses split transform merge idea and residual links

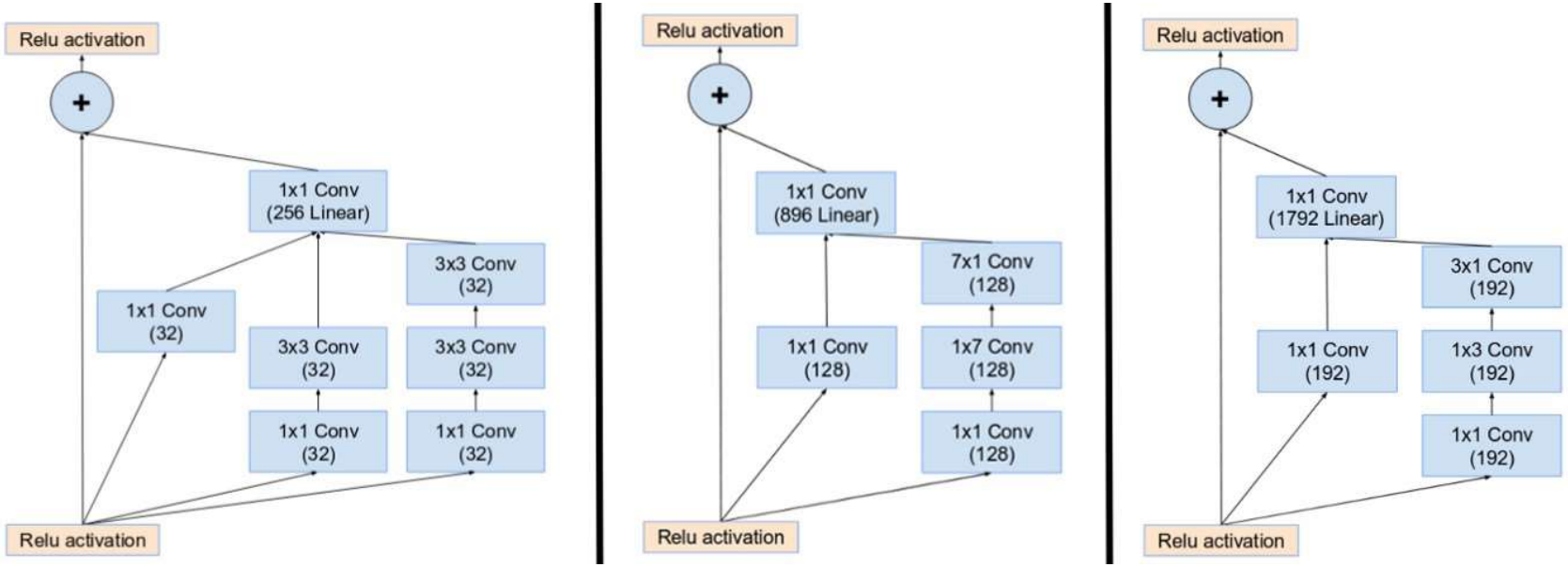


STEM

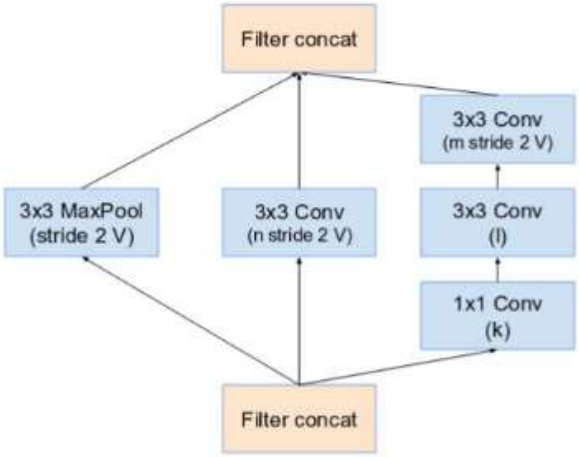


Inception Resnet Conncetion

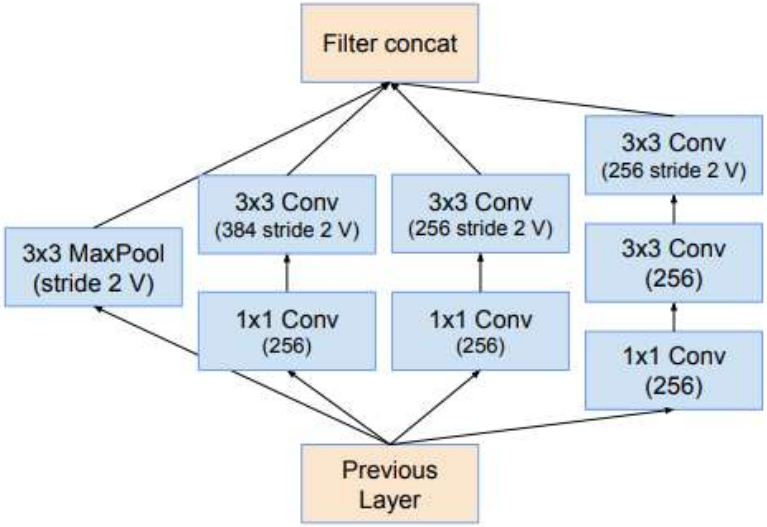




Inception modules A, B, C of Inception ResNet V1



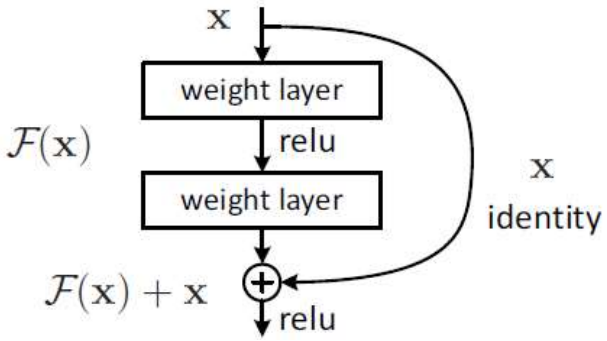
Reduction A schema



Inception ResNet-v1 Reduction Block B

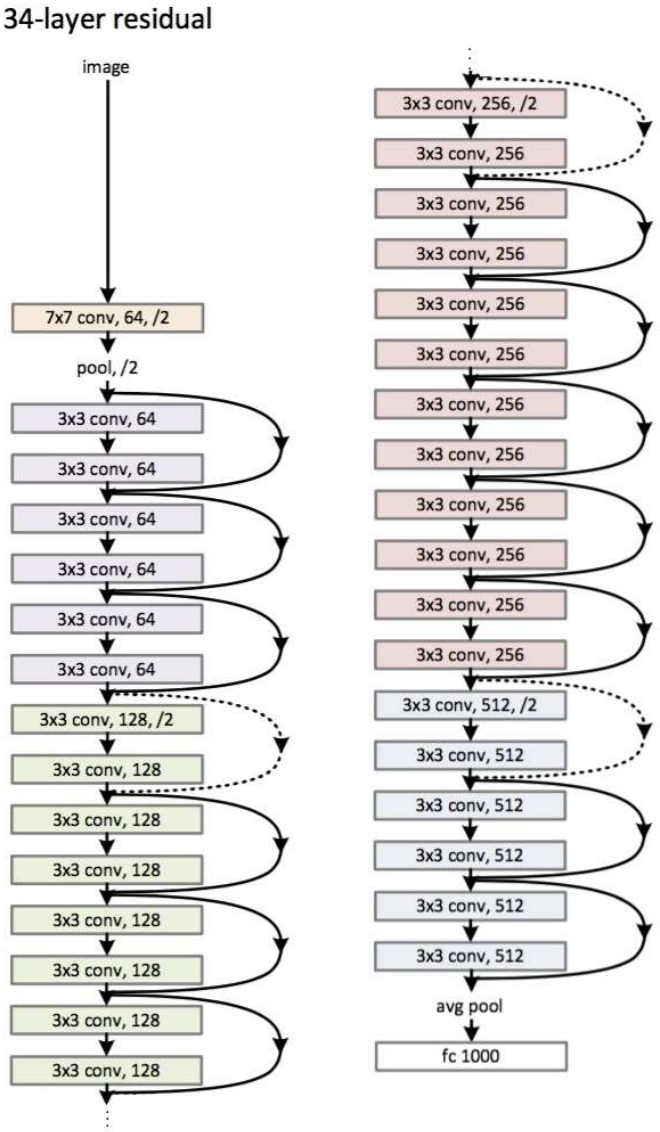
Resnet

- Residual learning
- Identity mapping based skip connections



Residual Blocks

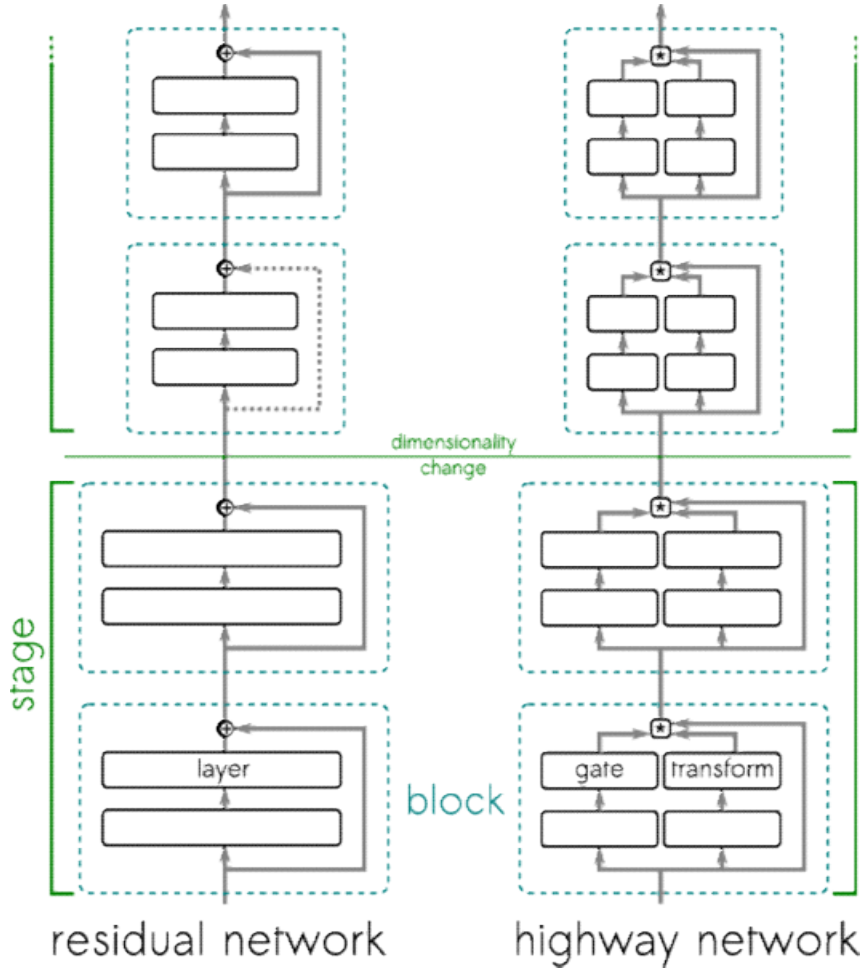
34-layers residual



Highway Network*

- Introduced an idea of Multi-path

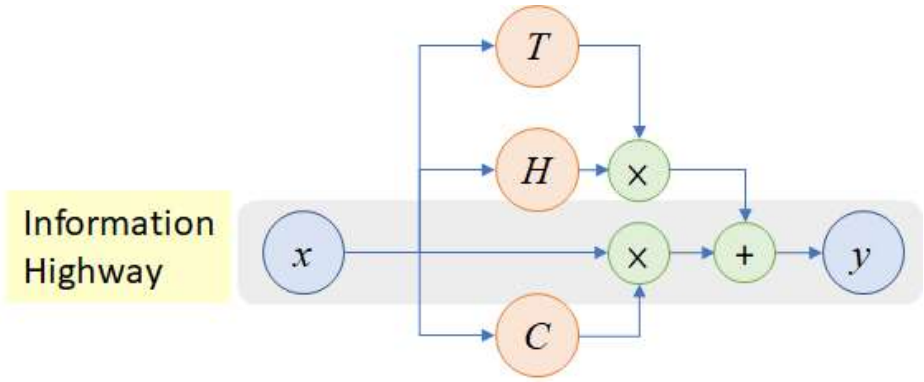
- In machine learning, a highway network is **an approach to optimizing networks and increasing their depth**.
- Highway networks use learned gating mechanisms to regulate information flow, inspired by Long Short-Term Memory (LSTM) recurrent neural networks.
- Highway networks have been used as part of text sequence labeling and speech recognition tasks



*Srivastava, Rupesh Kumar; Greff, Klaus; Schmidhuber, Jürgen (2 May 2015). "Highway Networks". arXiv:1505.00387 [cs.LG].

Continuing about High way networks

The model has two gates in addition to the **$H(W_H, x)$** gate: the transform gate **$T(W_T, x)$** and the carry gate **$C(W_C, x)$**. Those two last gates are non-linear transfer functions (by convention Sigmoid function). The **$H(W_H, x)$** function can be any desired transfer function.
The carry gate is defined as **$C(W_C, x) = 1 - T(W_T, x)$**. While the transform gate is just a gate with a sigmoid transfer function.



The structure of a hidden layer follows the equation:

$$y = H(x, W_H) \cdot T(x, W_T) + x \cdot C(x, W_C) = H(x, W_H) \cdot T(x, W_T) + x \cdot (1 - T(x, W_T))$$

The advantage of a Highway Network over the common deep neural networks is that solves or partially prevents the Vanishing gradient problem, thus leading to easier to optimize neural networks.

Table 5b Major challenges associated with implementation of Depth based CNN architectures.

Depth	With the increase in depth, the network can better approximate the target function with a number of nonlinear mappings and improved feature representations. Main challenge faced by deep architectures is the problem of vanishing gradient and negative learning.	
Architecture	**Strength**	**Gaps**
Inception-V3	• Exploited asymmetric filters and bottleneck layer to lessen the computational cost of deep architectures	• Complex architecture design • Lack of homogeneity
Highway Networks	• Introduced training mechanism for deep networks • Used auxiliary connections in addition to direct connections	• Parametric gating mechanism, difficult to implement
Inception-ResNet	• Combined the power of residual learning and inception block	-
Inception-V4	• Deep hierarchies of features, multilevel feature representation	• Slow in learning
ResNet	• Decreased the error rate for deeper networks • Introduced the idea of residual learning • Alleviates the effect of vanishing gradient problem	• A little complex architecture • Degrades information of feature-map in feed forwarding • Over adaption of hyper-parameters for specific task, due to the stacking of same modules

(3) Multi-Path based CNNs

CNNs which improve themselves by using skip connection cross layers to avoid information missing and gradient vanishing

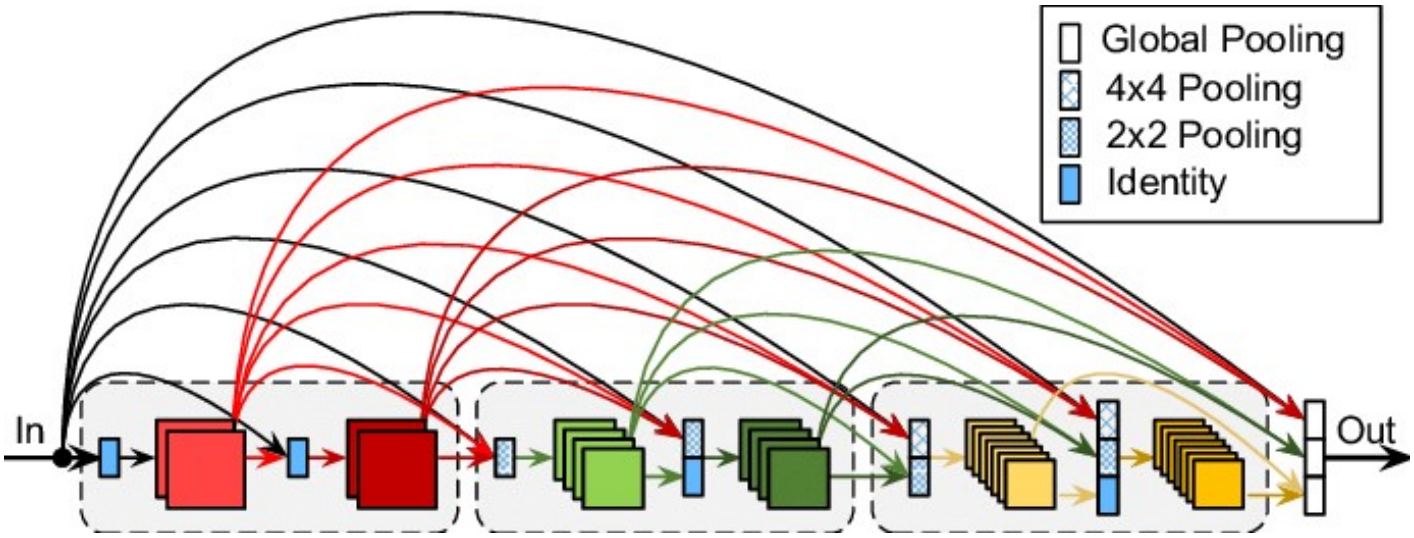
CNNs: Highway Networks ResNet DenseNet

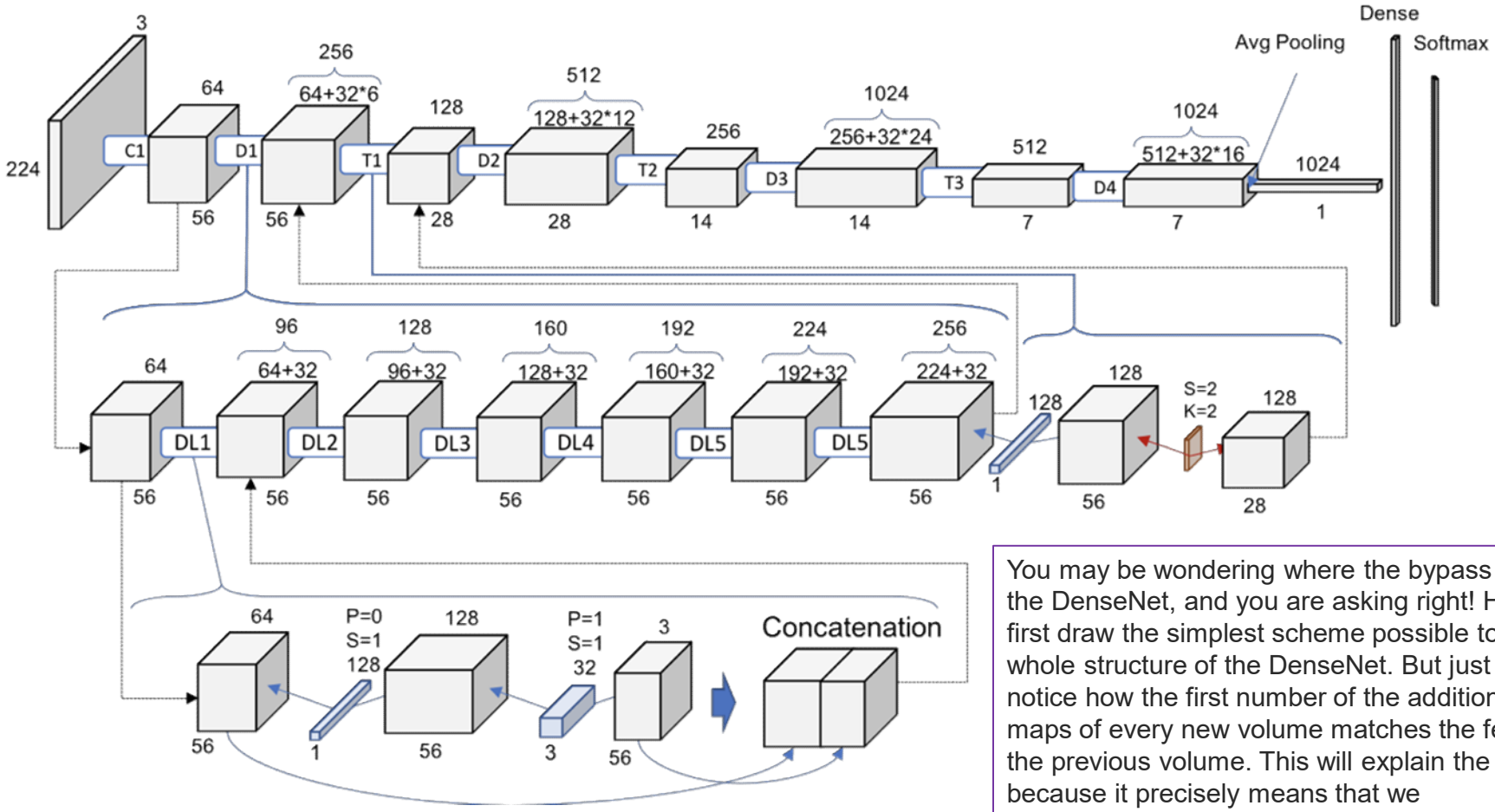
Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
Highway Networks	2015	- Introduced an idea of Multi-path	2.3 M	CIFAR-10: 7.76	19	Depth + Multi-Path	(Srivastava et al. 2015a)
ResNet	2016	- Residual learning - Identity mapping based skip connections	25.6 M 1.7 M	ImageNet: 3.6 CIFAR-10: 6.43	152 110	Depth + Multi-Path	(He et al. 2015a)
DenseNet	2017	- Cross-layer information flow	25.6 M 25.6 M 15.3 M 15.3 M	CIFAR-10+: 3.46 CIFAR100+:17.18 CIFAR-10: 5.19 CIFAR-100: 19.64	190 190 250 250	Multi-Path	(Huang et al. 2017)

DenseNet

- Cross-layer information flow

A problem with very deep networks was the problems to train, because of the mentioned flow of information and gradients.
DenseNets solve this issue since ***each layer has direct access to the gradients from the loss function*** and the original input image.





Full schematic representation of ResNet-121

You may be wondering where the bypass connections are all over the DenseNet, and you are asking right! However, I just wanted to first draw the simplest scheme possible to know go deeper on the whole structure of the DenseNet. But just for the curious, you can notice how the first number of the addition to calculate the feature maps of every new volume matches the feature maps dimension of the previous volume. This will explain the bypass connection because it precisely means that we are **concatenating** (concatenate means add dimension, but not add values!) **new information to the previous volume**, which is being **reused**.

Table 5c Major challenges associated with implementation of Multi-Path based CNN architectures.

Multi-Path	Shortcut paths provides the option to skip some layers. Different types of the shortcut connections used in literature are zero padded, projection, dropout, 1x1 connections, etc.	
Architecture	**Strength**	**Gaps**
Highway Networks	• Mitigates the limitations of deep networks by introducing cross layer connectivity.	• Gates are data dependent and thus may become parameter expensive
ResNet	• Use of identity based skip connections to enable cross layer connectivity • Information flow gates are data independent and parameter free • Can easily pass the signal in both directions, forward and backward	• Many layers may contribute very little or no information • Relearning of redundant feature-maps may happen
DenseNet	• Introduced depth or cross-layer dimension • Ensures maximum data flow between the layers in the network • Avoid relearning of redundant feature-maps • Low and high level both features are accessible to decision layers	• Large increase in parameters due to increase in number of feature-maps at each layer

Width based Multi-Connection CNNs

CNNs which improve themselves by making parallel use of multiple processing units within a layer
Parallel and homogenous topology in each block

CNNs: Wide ResNet Pyramidal Net Xception ResNeXt Inception Family

Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
WideResNet	2016	- Width is increased and depth is decreased	36.5 M	CIFAR-10: 3.89 CIFAR-100: 18.85	28 -	Width	(Zagoruyko and Komodakis 2016)
PyramidalNet	2017	- Increases width gradually per unit	116.4 M 27.0 M 27.0 M	ImageNet: 4.7 CIFAR-10: 3.48 CIFAR-100: 17.01	200 164 164	Width	(Han et al. 2017)
Xception	2017	- Depth wise convolution followed by point wise convolution	22.8 M	ImageNet: 0.055	126	Width	(Chollet 2017)
ResNeXt	2017	- Cardinality - Homogeneous topology - Grouped convolution	68.1 M	CIFAR-10: 3.58 CIFAR-100: 17.31 ImageNet: 4.4	29 - 101	Width	(Xie et al. 2017)
Inception-V3	2015	- Handles the problem of a representational bottleneck - Replace large size filters with small filters	23.6 M	ImageNet: 3.5 Multi-Crop: 3.58 Single-Crop: 5.6	159	Depth + Width	(Szegedy et al. 2016b)
Inception-V4	2016	- Split transform and merge idea Uses asymmetric filters	35 M	ImageNet: 4.01	70	Depth +Width	(Szegedy et al. 2016a)
Inception-ResNet	2016	- Uses split transform merge idea and residual links	55.8M	ImageNet: 3.52	572	Depth + Width + Multi-Path	(Szegedy et al. 2016a)

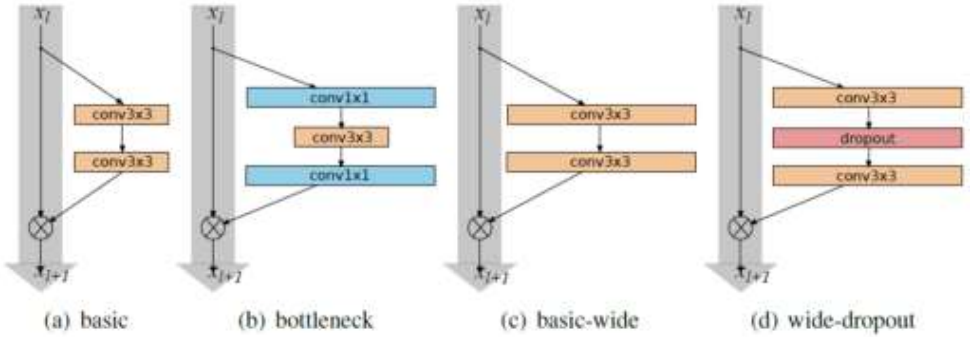
Wide ResNet

a variant of Resent to reduce the depth and increase the width

In WRNs, plenty of parameters are tested such as the design of the ResNet block, how deep (deepening factor *l*) and how wide (widening factor *k*) within the ResNet block.

When *k*=1, it has the same width of *ResNet*. While *k*>1, it is *k* time wider than *ResNet*.

WRN-*d*-*k*: means the WRN has the depth of *d* and with widening factor *k*.

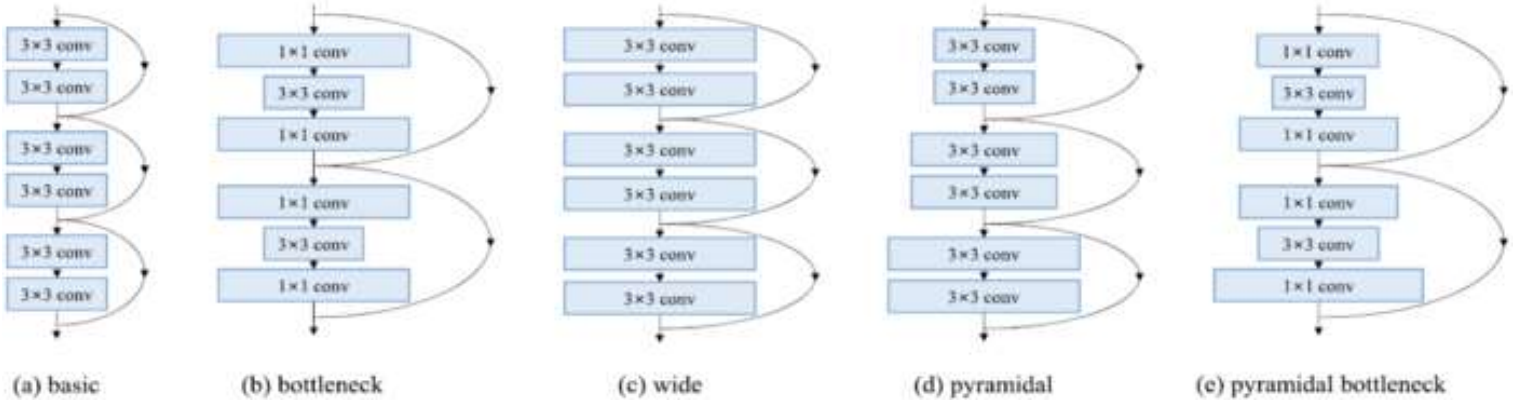


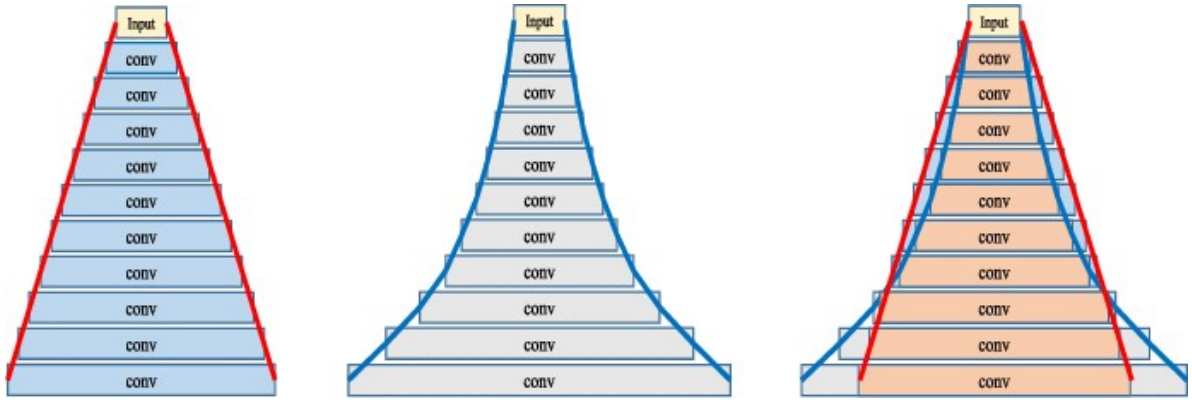
(a) basic (b) bottleneck (c) basic-wide (d) wide-dropout

	depth-*k*	# params	CIFAR-10	CIFAR-100
NIN [20]			8.81	35.67
DSN [19]			8.22	34.57
FitNet [24]			8.39	35.04
Highway [28]			7.72	32.39
ELU [5]			6.55	24.28
original-ResNet[11]	110	1.7M	6.43	25.16
	1202	10.2M	7.93	27.82
stoc-depth[14]	110	1.7M	5.23	24.58
	1202	10.2M	4.91	-
pre-act-ResNet[13]	110	1.7M	6.37	-
	164	1.7M	5.46	24.33
	1001	10.2M	4.92(4.64)	22.71
WRN (ours)	40-4	8.9M	4.53	21.18
	16-8	11.0M	4.27	20.43
	28-10	36.5M	**4.00**	**19.25**

Pyramidal Net

Pyramidal Net increases the width gradually per residual unit.





Xception

- Depth wise convolution followed by point wise convolution

Xception modified the original inception block by making it wider and replacing the different spatial dimensions (1x1, 5x5, 3x3) with a single dimension (3x3) followed by a 1x1 convolution to regulate computational complexity.

Xception makes the network computationally efficient by decoupling spatial and feature-map (channel) correlation,

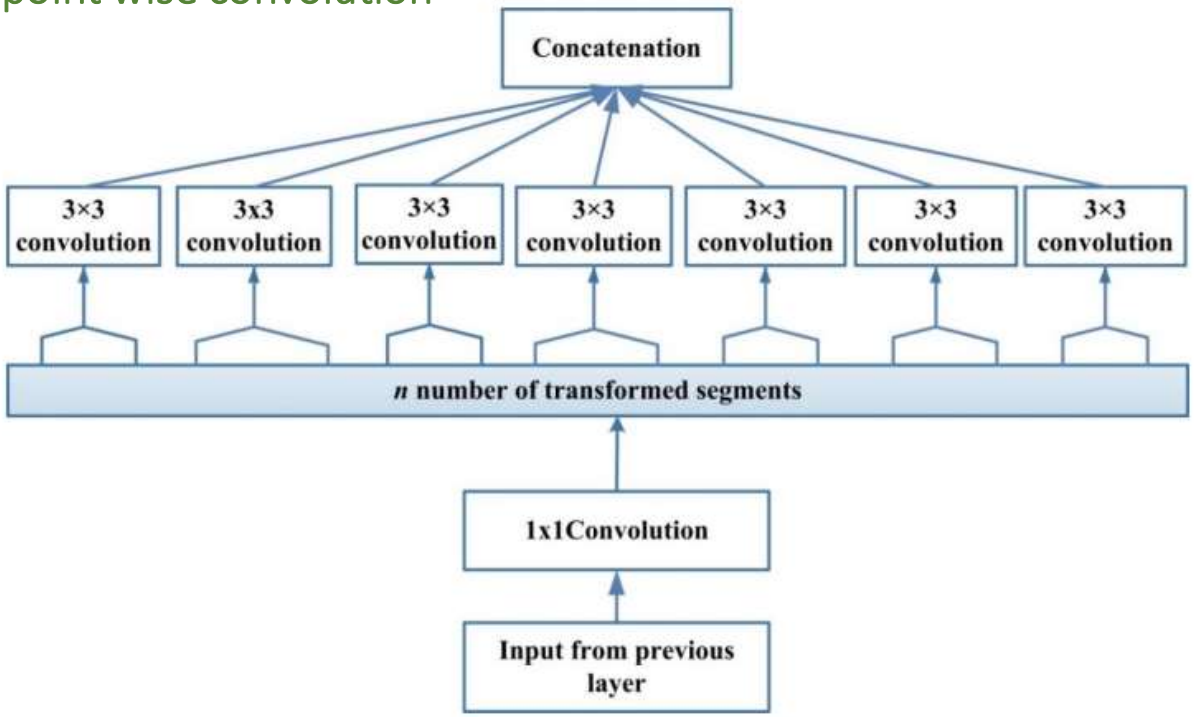


Fig. 8 Xception building block and its *n* sets of transformation.

ResNeXt

-Cardinality
- Homogeneous topology
- Grouped convolution

ResNeXt, also known as Aggregated Residual Transform Network, is an improvement over the Inception Network (Xie et al. 2017).
Xie et al. exploited the concept of the split, transform, and merge in a powerful but simple way by introducing a new term; cardinality (Szegedy et al. 2015).
Cardinality is an additional dimension, which refers to the size of the set of transformations (Han et al. 2018; Sharma and Muttoo 2018).

Refer to the following figure, the architecture includes 32 same topology blocks so the value of cardinality is 32.

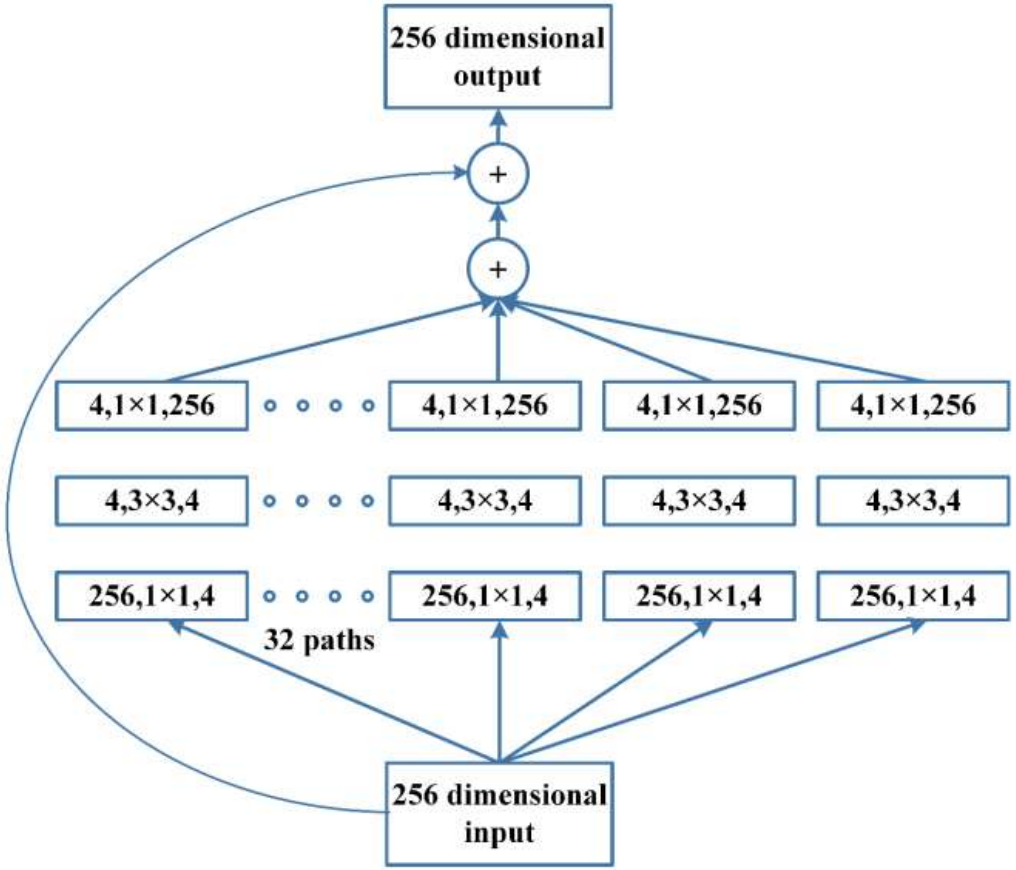


Fig. 9 ResNeXt building block showing the different paths of transformation.

Table 5d Major challenges associated with implementation of Width based CNN architectures.

Width	Earlier, it was assumed that to improve accuracy, the number of layers have to be increased. However, by increasing the number of layers, the vanishing gradient problem arises and training might get slow. So, the concept of widening a layer was also investigated.	
Architecture	**Strength**	**Gaps**
Wide ResNet	• Showes the effectiveness of parallel use of transformations by increasing the width of ResNet and decreasing its depth • Enables feature reuse • Have shown that dropouts between the convolutional layer are more effective	• Over fitting may occur • More parameters than thin deep networks
Pyramidal Net	• Introduces the idea of increasing the width gradually per unit • Avoids rapid information loss • Covers all possible locations instead of maintaining the same dimension till last unit	• High spatial and time complexity • May become quite complex, if layers are substantially increased
Xception	• Introduce the concept that learning across 2D followed by 1 D is easier than to learn filters in 3 D space • Depth-wise separable convolution is introduced • Use of cardinality to learn good abstractions	• High computational cost
Inception	• Varying size filters inside inception module increases the output of the intermediate layers • Varying size filters are helpful to capture the diversity in high-detail images	• Increase in space and time complexity
ResNeXt	• Introduced cardinality to avail diverse transformations at each layer • Easy parameter customization due to homogenous topology • Uses grouped convolution	• High computational cost

(4) Feature-Map (Channel *FMap*) Exploitation based CNNs

CNNs which improve themselves by adding a block for the selection of feature-maps (channels)

CNNs: Squeeze and Excitation Network, Competitive Squeeze and Excitation Networks

Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
Squeeze & Excitation Networks	2017	- Models interdependencies between feature-maps	27.5 M	ImageNet: 2.3	152	Feature-Map Exploitation	(Hu et al. 2018a)
Competitive Squeeze & Excitation Network CMPE-SE-WRN-28	2018	- Residual and identity mappings both are used for rescaling the feature-map	36.92 M 36.90 M	CIFAR-10: 3.58 CIFAR-100: 18.47	152 152	Feature-Map Exploitation	(Hu et al. 2018b)

(5) Squeeze and Excitation Network

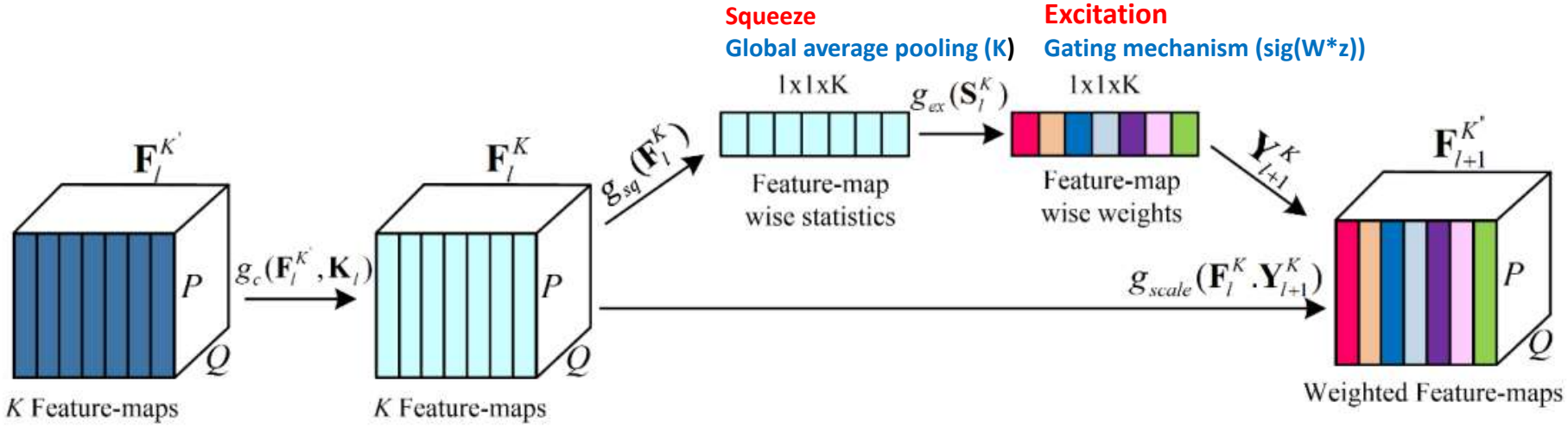
Model interdependencies between feature maps

Squeeze and Excitation block showing the computation of masks for the recalibration of feature-

maps that are commonly known as channels in literature

Squeeze-and-Excitation Networks (SENets) **introduce a building block for CNNs that improves channel interdependencies at almost no computational cost**. ...

Let's add parameters to each channel of a convolutional block so that the network can adaptively adjust the weighting of each feature map



Competitive Squeeze and Excitation Networks

*Hu et al. 2018b

- The authors used the idea of SEblock to improve the learning of deep residual networks .
- SE-Network recalibrates the feature-maps based upon their contribution in class discrimination. However, the main concern with SE-Net is that in ResNet, it only considers the residual information for determining the weight of each feature-map (Hu et al. 2018a).
- This minimizes the impact of SEblock and makes ResNet information redundant.
- Hu et al. addressed this problem by generating feature-map wise motifs (statistics) from both residual and identity mapping based feature-maps.
- In this regard, global representation of feature-maps is generated using global average pooling
  operation, whereas relevance of feature-maps is estimated by establishing competition between feature descriptors of residual and identity mappings.
- This phenomena is termed as inner imaging. CMPE-SE block not only models the relationship between residual feature-maps but also maps their relation with identity feature-map.

Competitive Squeeze-Excitation Architecture for Residual block

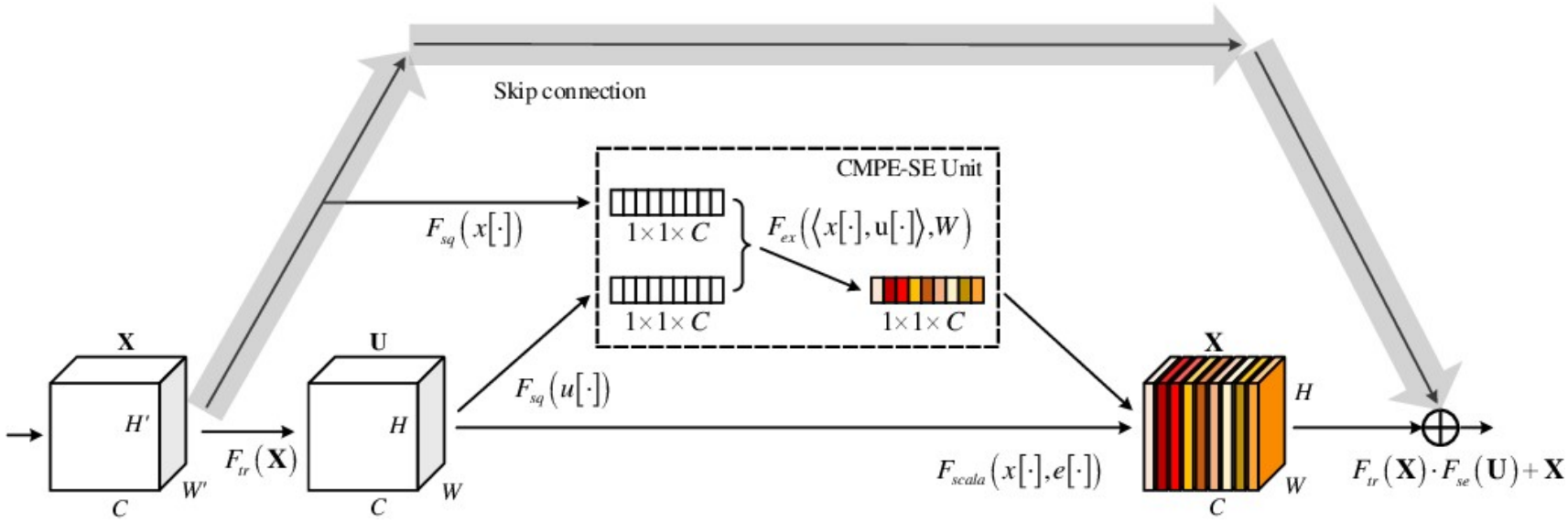


Table 5e Major challenges associated with implementation of Feature-Map exploitation based CNN architectures.

Feature-Map Selection	As the deep learning topology is extended, more and more features maps are generated at each step. Many of the Feature-maps might be important for classification task, others might redundant or less important. Hence, feature-map selection is another important dimension in deep learning architectures.	
Architecture	**Strength**	**Gaps**
Squeeze and Excitation Network	• It is a block-based concept • Introduced a generic block that can be added easily in any CNN model due to its simplicity • Squeezes less important features and vice versa	• In ResNet, it only considers the residual information for determining the weight of each channel
Competitive Squeeze and Excitation Networks	• Uses feature-map wise statistics from both residual and identity mapping based features • Makes a competition between residual and identity feature-maps	• Doesn't support the concept of attention

(6) Channel(*Input*) Exploitation based CNNs

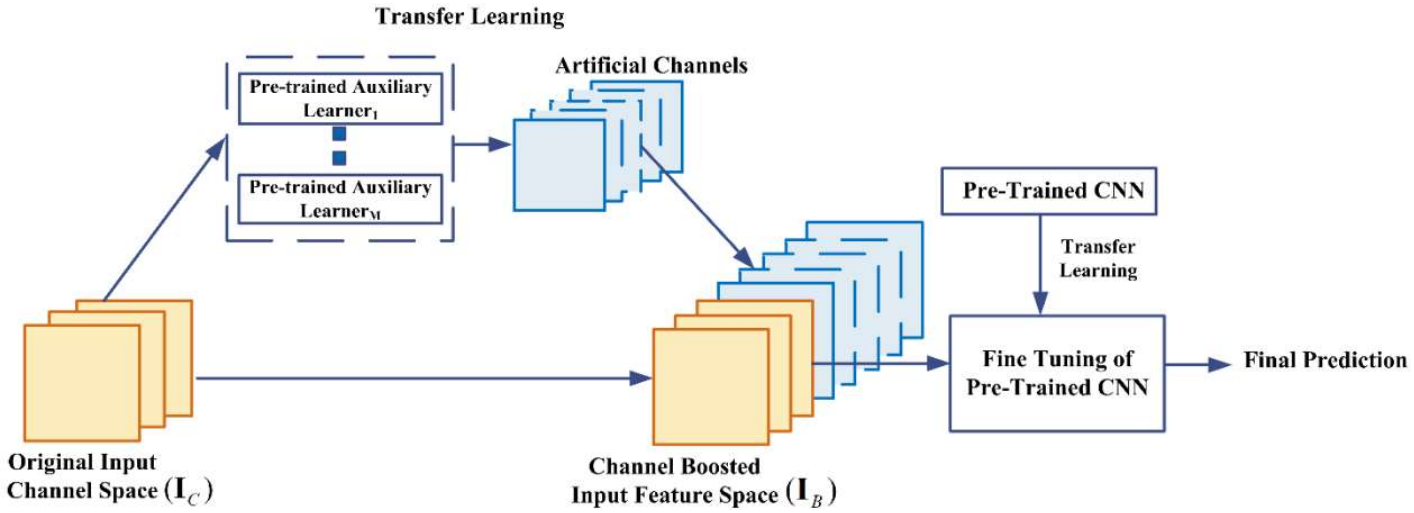
CNNs which improve themselves by using auxiliary learners for channel boosting (input channel dimension)

CNNs: Channel Boosted CNN using TL

Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
Channel Boosted CNN	2018	- Boosting of original channels with additional information rich generated artificial channels	-	-	-	Channel Boosting	(Khan et al. 2018a)

Channel Boosted CNN using TL (transfer Learning)

Basic architecture of CB-CNN showing the deep auxiliary learners for creating artificial channels



In the proposed methodology, **a deep CNN is boosted by various channels available through TL from already trained Deep Neural Networks, in addition to its original channel**. The deep architecture of CNN then exploits the original and boosted channels down the stream for learning discriminative patterns.

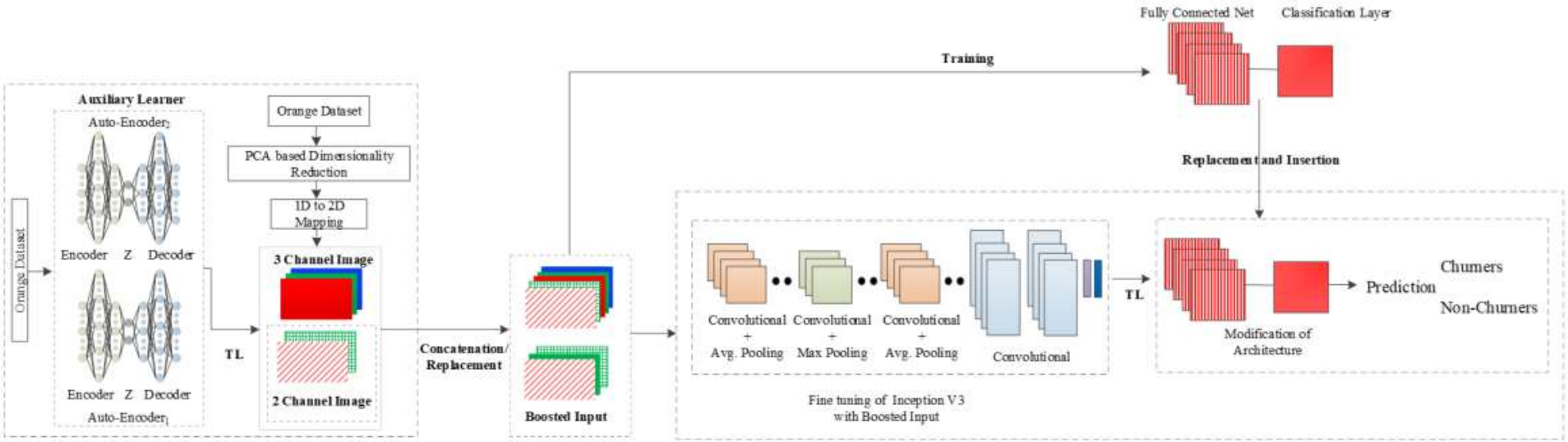


Fig. 4. Details of the working of the proposed CB-CNN. *CB-CNN1* is trained by using Boosted Input$_1$. Boosted Input$_1$ is comprised of three channels that is generated by replacing the input channels of original feature space with two auxiliary channels. Whereas, Boosted Input$_2$ is used to train *CB-CNN2* that is generated by concatenating original channel space with three auxiliary channels.

Table 5f Major challenges associated with implementation of Channel Boosting based CNN architectures.

Channel Boosting	The learning of CNN also relies on the input representation. The lack of diversity and absence of class discernable information in the input may affect CNN performance. For this purpose, the concept of channel boosting (input channel dimension) using auxiliary learners is introduced in CNN to boost the representation of the network (Khan et al. 2018a).	
Architecture	**Strength**	**Gaps**
Channel Boosted CNN using Transfer Learning	• It boosts the number of input channels for improving the representational capacity of the network • Inductive Transfer Learning is used in a novel way to build a boosted input representation for CNN	• Increases in computational load may happen due to the generation of auxiliary channels

(7) Attention based CNNs

CNNs which use attention mechanism to pay attention to context-relevant parts

CNNs: Residual Attention Neural Network Convolutional Block Attention Module Concurrent Spatial and Channel Excitation Mechanism

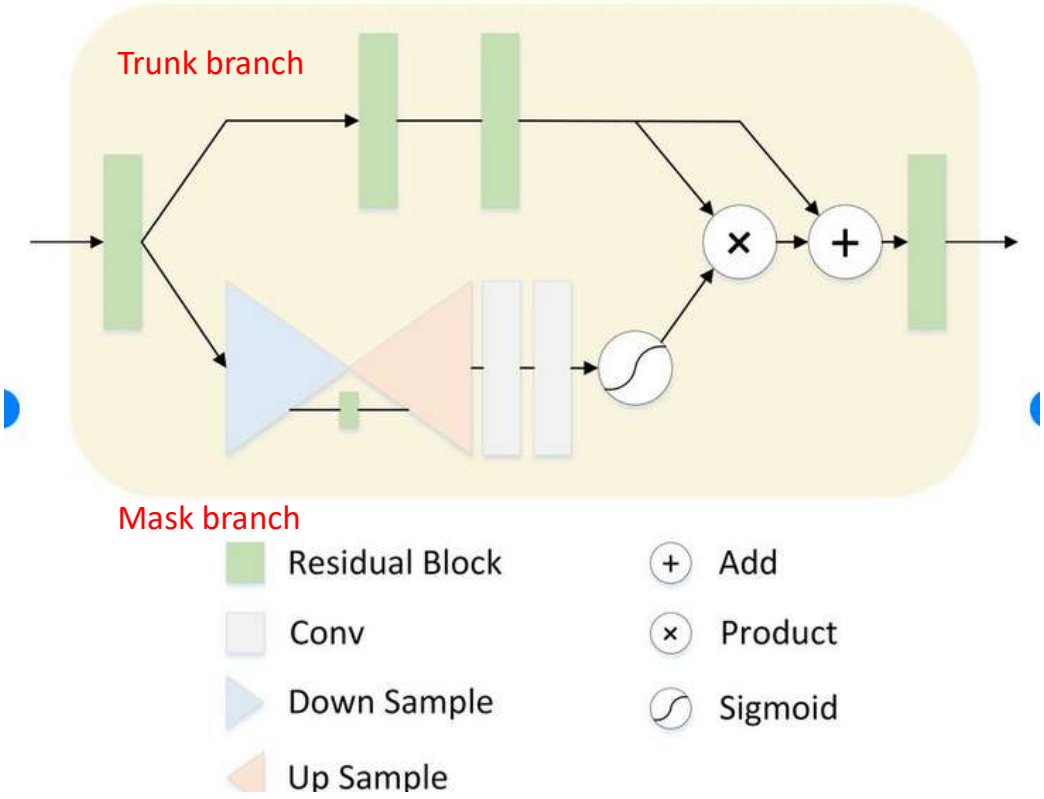
Architecture Name	Year	Main contribution	Parameters	Error Rate	Depth	Category	Reference
Residual Attention Neural Network	2017	- Introduced an attention mechanism	8.6 M	CIFAR-10: 3.90 CIFAR-100: 20.4 ImageNet: 4.8	452	Attention	(Wang et al. 2017a)
Convolutional Block Attention Module (ResNeXt101 (32x4d) + CBAM)	2018	- Exploits both spatial and feature-map information	48.96 M	ImageNet: 5.59	101	Attention	(Woo et al. 2018)
Concurrent Spatial & Channel Excitation Mechanism	2018	- Spatial attention - Feature-map attention - Concurrent placement of spatial and channel attention	-	MALC: 0.12 Visceral: 0.09	-	Attention	(Roy et al. 2018)

Residual Attention Neural Network*

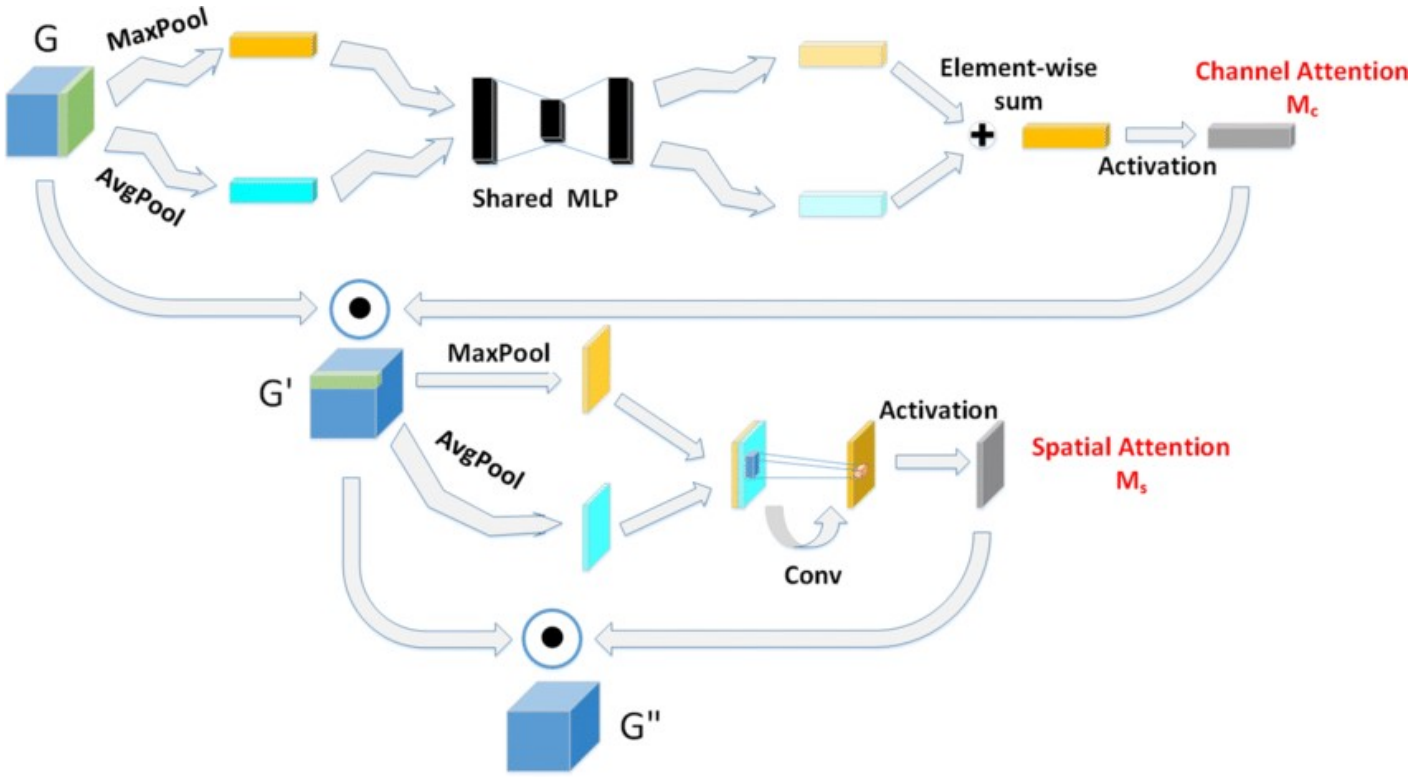
Sik-Ho Tsang Apr 11, 2019

Multiple attention module is stacked to generate attention-aware features. Attention residual learning is used for very deep network.

Attention module. The top branch is the trunk branch that consists of two residual blocks. The bottom branch is the mask branch.

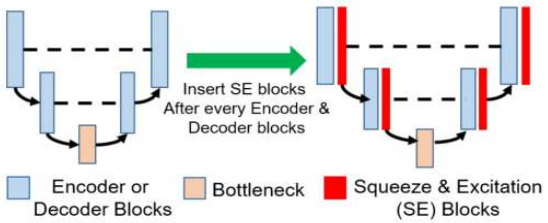


CBAM: Convolutional Block Attention Module

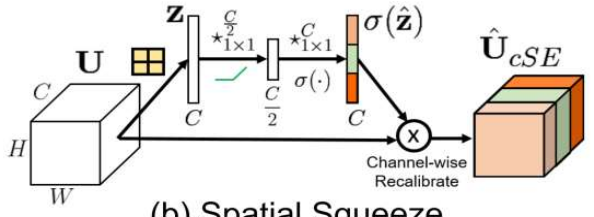


Concurrent Spatial and Channel Excitation Mechanism

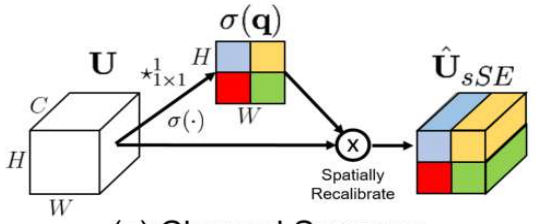
F-CNN: Fully Convolutional NN.



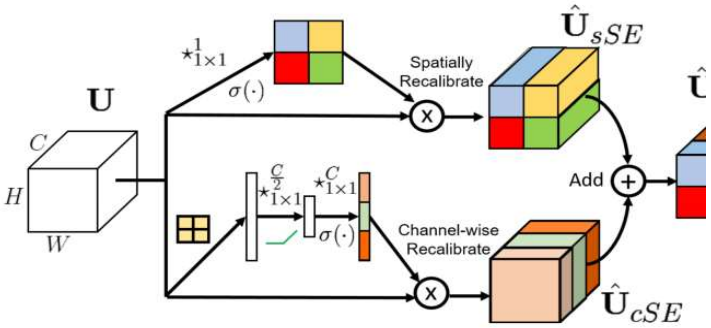
(a) SE block in F-CNN



(b) Spatial Squeeze
and Channel Excitation (cSE)



(c) Channel Squeeze
and Spatial Excitation (sSE)



(d) Concurrent Spatial and Channel
Squeeze and Channel Excitation (scSE)

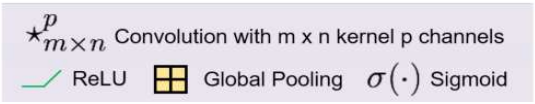


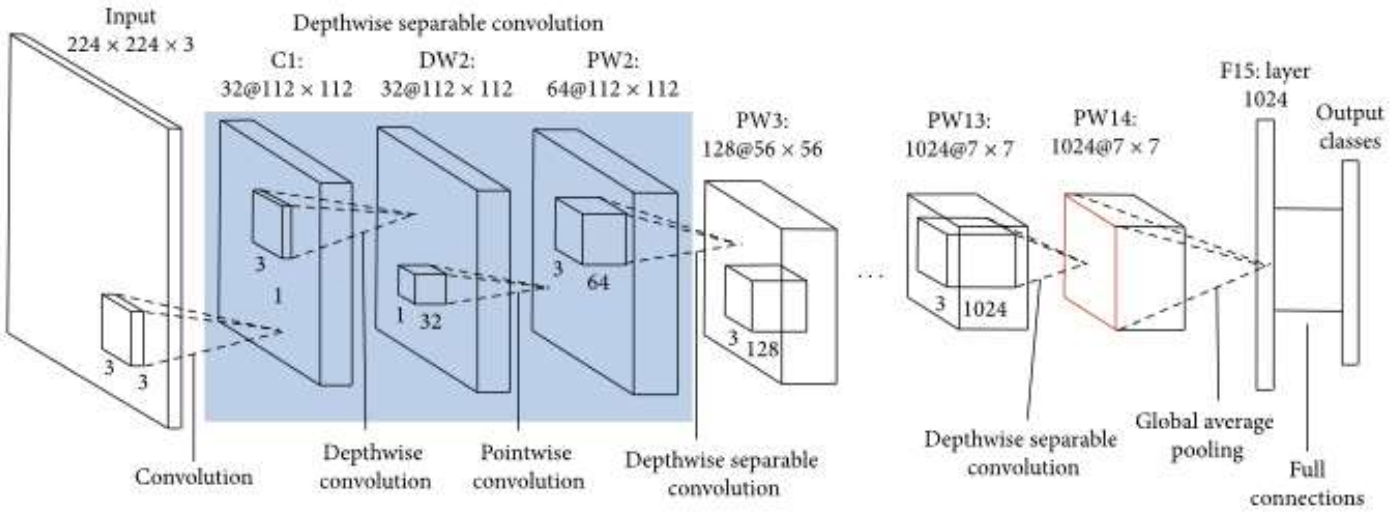
Table 5g Major challenges associated with implementation of Attention based CNN architectures.

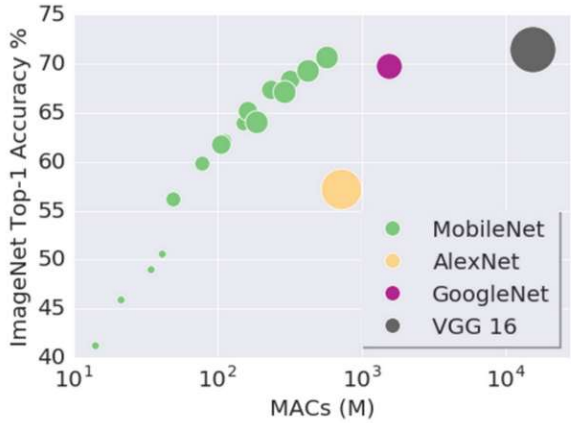
Attention	Attention Networks advantages to choose which patch is the area of the focus or most important in an image	
Architecture	**Strength**	**Gaps**
Residual Attention Neural Network	• Generates attention aware feature-maps • Easy to scale up due to residual learning • Provides different representations of the focused patches • Adds soft weights on features using bottom up top-down feedforward attention	• Complex model
Convolutional Block Attention Module	• CBAM is a generic block designed for feed forward convolutional neural networks. • Generate both feature-map and spatial attention in a sequential manner • Channel attention maps help what to focus. • Spatial attention helps where to focus. • Increases efficient flow of information. • Uses global average pooling and max pool simultaneously.	• Increase in computational load may happen

Mobile Net

it uses depthwise separable convolutions to build lightweight deep neural networks

What is MobileNet?
MobileNet is **a type of convolutional neural network designed for mobile and embedded vision applications**. They are based on a streamlined architecture that uses depthwise separable convolutions to build lightweight deep neural networks that can have low latency for mobile and embedded devices.



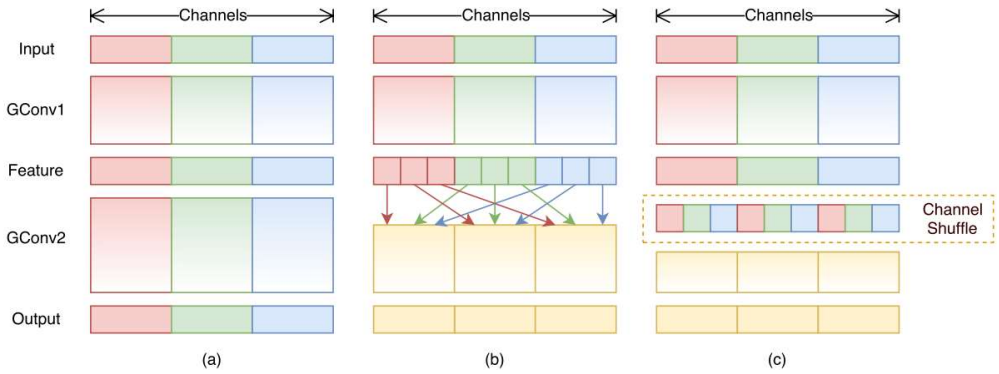


The speed and power consumption of the network is proportional to the number of MACs (Multiply-Accumulates) which is a measure of the number of fused Multiplication and Addition operation

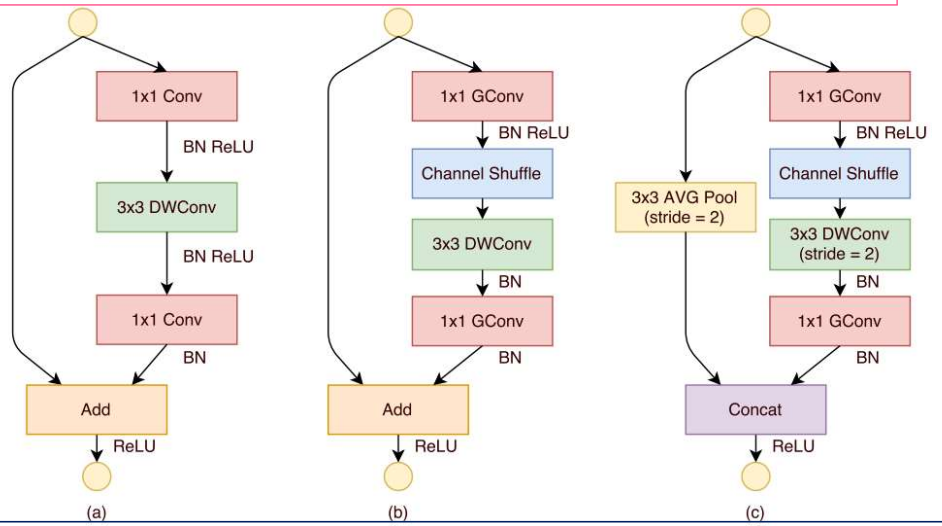
ShuffleNet

An Extremely Efficient Convolutional Neural Network for Mobile

The ShuffleNet utilizes pointwise group convolution and channel shuffle **to reduce computation cost while maintaining accuracy**. It manages to obtain lower top-1 error than the MobileNet system on ImageNet classification, and achieves ~13x actual speedup over AlexNet while maintaining comparable accuracy



Channel shuffle with two stacked group convolutions. GConv stands for group convolution. a) No cross talk; b) GConv2 takes data from different groups after GConv1; c) an equivalent implementation to b) using channel shuffle.



ShuffleNet Units. a) bottleneck unit with depthwise convolution (DWConv) b) ShuffleNet unit with pointwise group convolution (GConv) and channel shuffle; c) ShuffleNet unit with stride = 2.

Efficient Net

EfficientNet is **a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compound coefficient**.

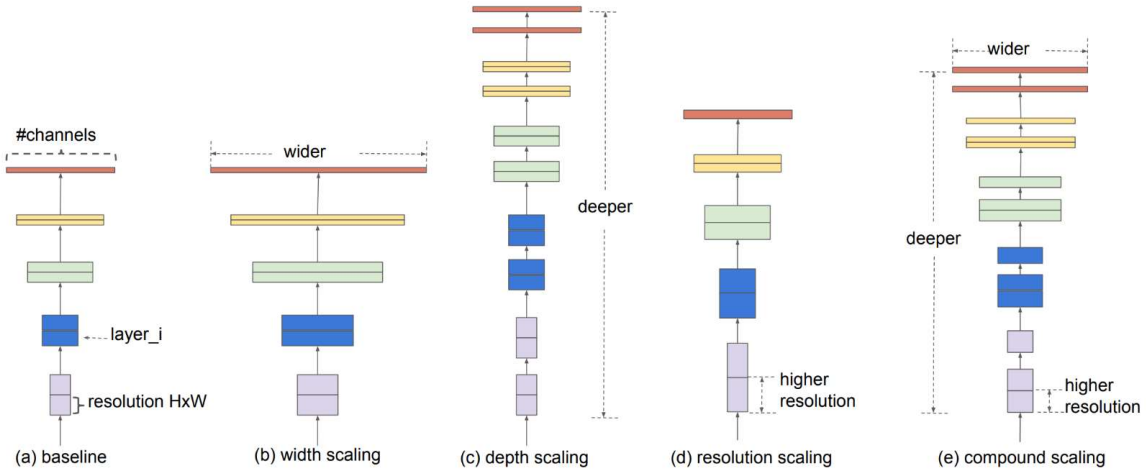
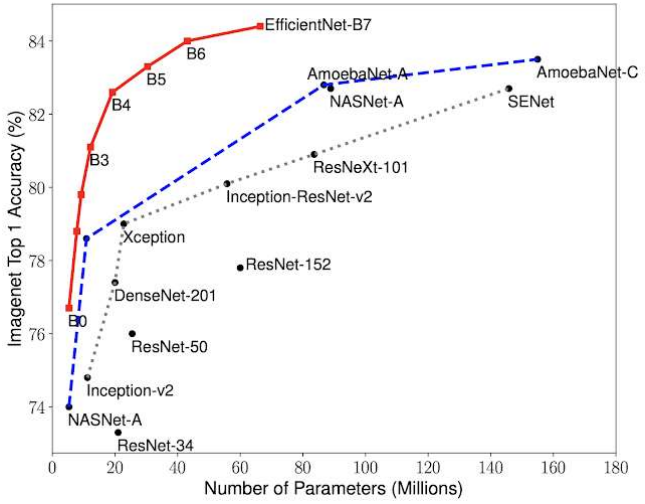


Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.



Model Size vs. Accuracy Comparison. EfficientNet-B0 is the baseline network developed, while Efficient-B1 to B7 are obtained by scaling up the baseline network.

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPS	Ratio-to-EfficientNet
EfficientNet-B0	**76.3%**	**93.2%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**78.8%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**79.8%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.1%**	**95.5%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.6%**	**96.3%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.3%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.9%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.4%**	**97.1%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

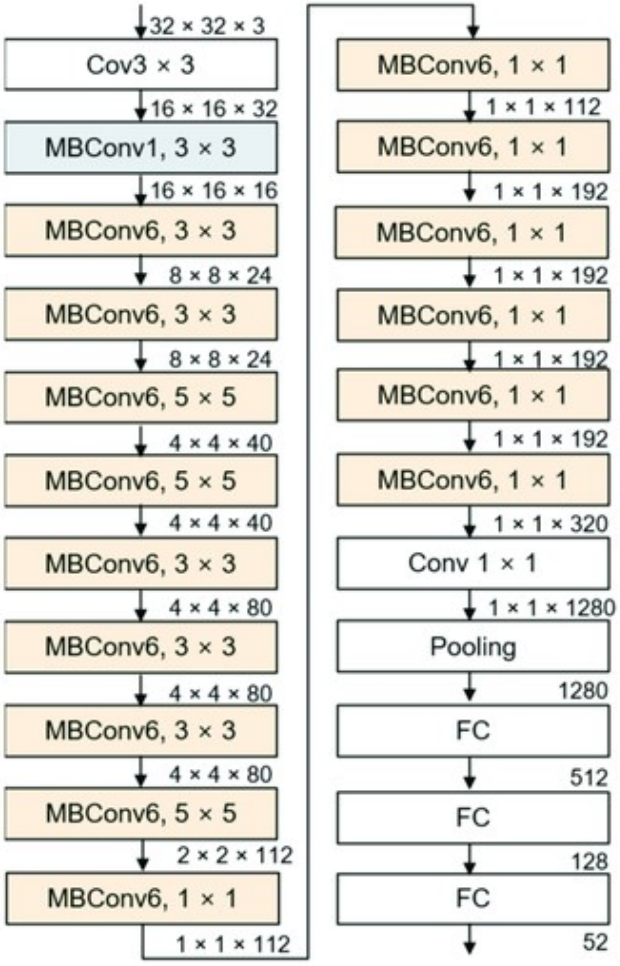
We omit ensemble and multi-crop models (Hu et al., 2018), or models pretrained on 3.5B Instagram images (Mahajan et al., 2018).

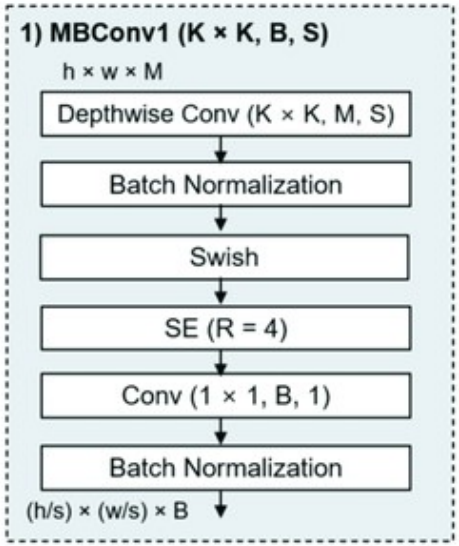
FLOPS = **floating point operations per second**

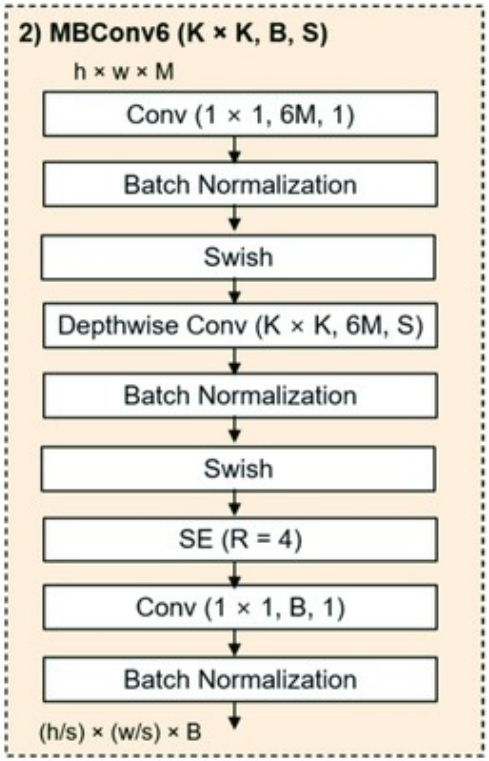
EfficientNetB0

The structure of an EfficientNetB0 model with the internal structure of MBConv1 and MBConv6. Compared to MBConv1, MBConv6 has three layers at the top. The number of feature maps as the output is 6.

A MBConv is **a Inverted Linear BottleNeck layer with Depth-Wise Separable Convolution** and with squeeze and excitation connection added to it.



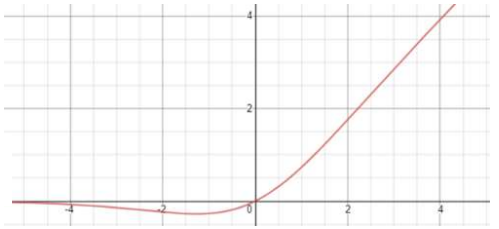




K : kernel size
M : Input Feature maps
B : Output Feature maps
S : Stride
R : Reduction ration of SE

Formally stated, the Swish activation function is…

$$f(x) = x * (1 + \exp(-x))^{-1}$$



3.2. Layer-wise Analysis Algorithm

The concept of dataflow or information-flow

- By applying the input data, $\boldsymbol{x}$ ($\boldsymbol{x} \equiv \boldsymbol{x}^1$) to a deep neural network with N_L layers, the data will be transformed layer by layer.
- Dataflow (information-flow) denotes the data which transform by layers of a deep neural network :

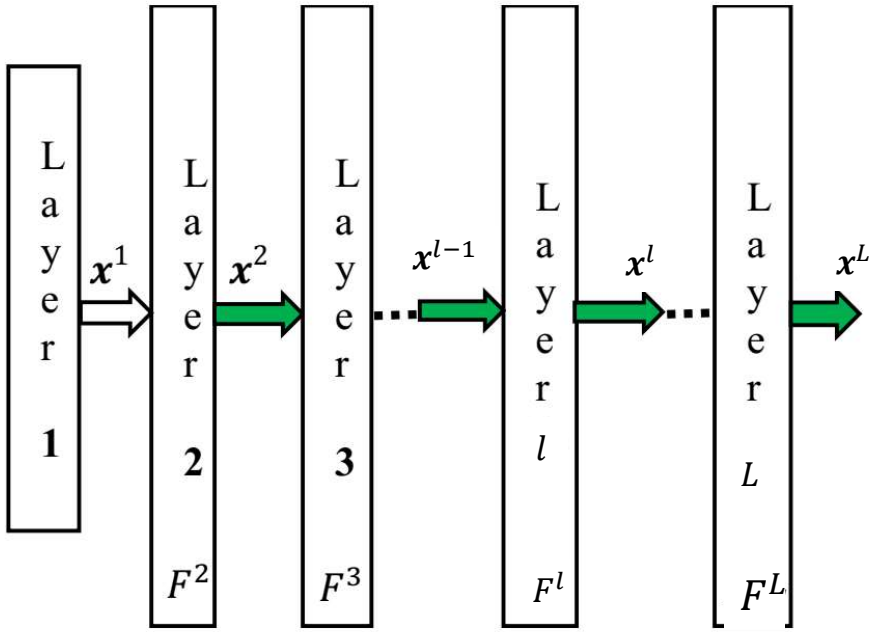
$$\boldsymbol{x}^1 \rightarrow \cdots \boldsymbol{x}^{l-1} \rightarrow \boldsymbol{x}^l \rightarrow \cdots \rightarrow x^L$$

- L: number of layers in the model
- $\boldsymbol{x}^l$ denotes the dataflow at layer l, which is reshaped as a vector and its length is n_L.

$$\boldsymbol{x}^l \boldsymbol{\epsilon} \boldsymbol{R}^{\boldsymbol{n_l} \times \boldsymbol{1}}$$

- One can compute SI(SmI) for dataflow at layer $\boldsymbol{l}$:

$Data^l = \{(\boldsymbol{x}_i^l, l_i)\}_{i=1}^m$ $\quad\quad$ $\boldsymbol{l}$=1,2, … , L



It seems the above DNN is a feedforward network.
However, each layer can be a RNN or a LSTM module, too.
In the case of RNN or LSTM, it is assumed that the hidden state is within the layer.

The complexity measure SI (SmI) in DNNs

Some definitions

- Disturbance: non relevant information which disturb the feature space
- Distortion: Uncertainties due to (1) inherent appearances of the features, (2)the environment constraints on the measuring process, and (3) different measuring parameters.
- Common features: In classification problems, features which are common between examples of different classes they avoid discrimination between different classes.
- Exclusive features: Features which discriminate different classes in a classification problem or maximize smoothness in regression problems.

Two important notes:

1. In a DNN, it is expected that after a certain number of filter layers, a feature space with negligible disturbance, distortion, and common features is appeared.
2. Using a metric formulation, it is proved that by intensifying exclusive features and weakening disturbances, distortions, and common features (complexity), the SI(SmI) of dataflow will increase gradually through layers of a DNN.

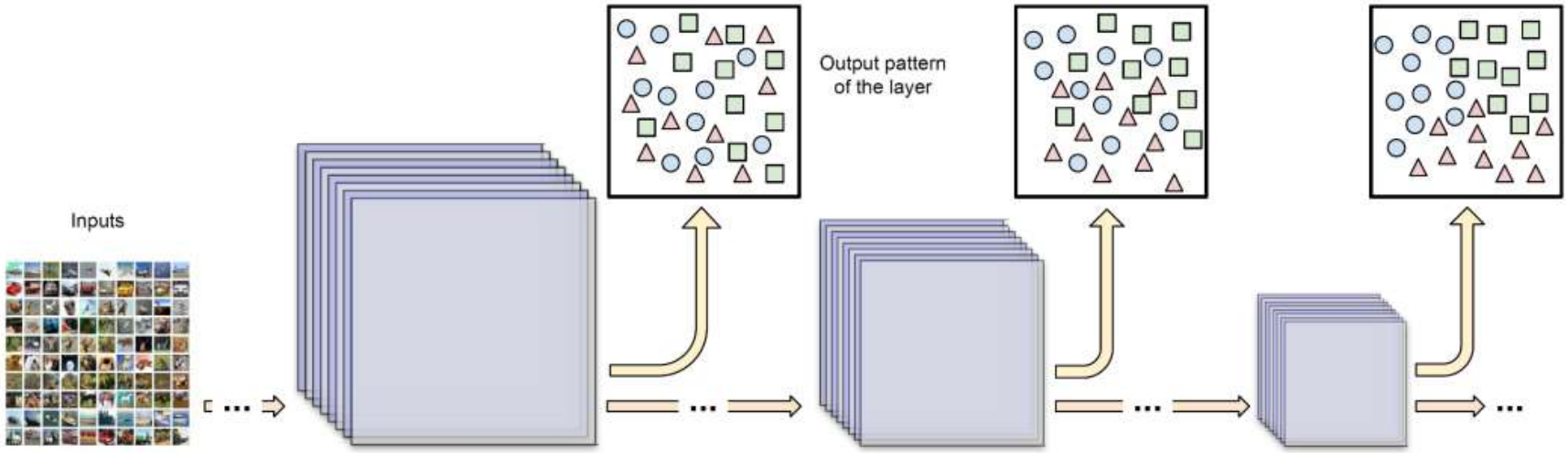
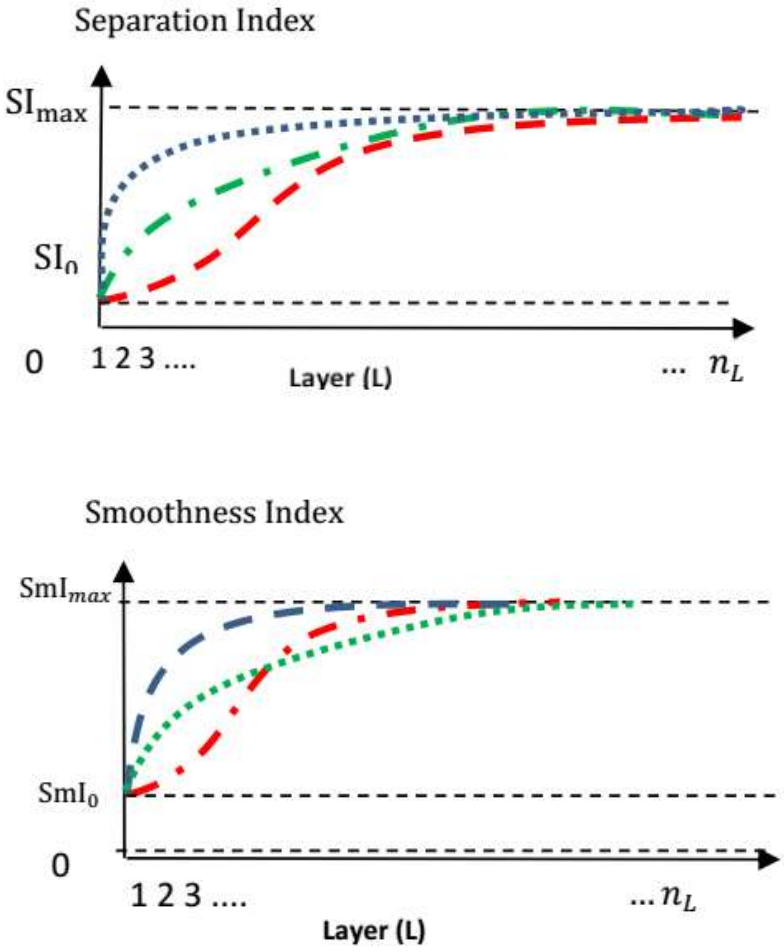


Figure 1. Output pattern of each layer through the network.

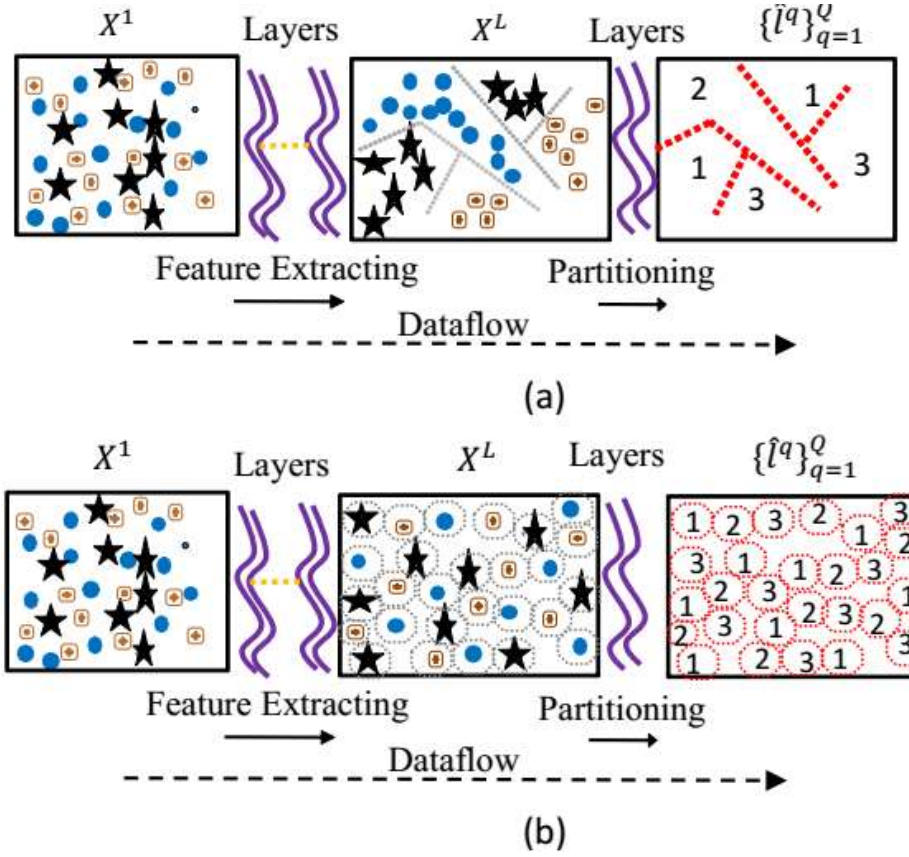
It is expected that the complexity of data decreases layer by layer in a deep neural network.

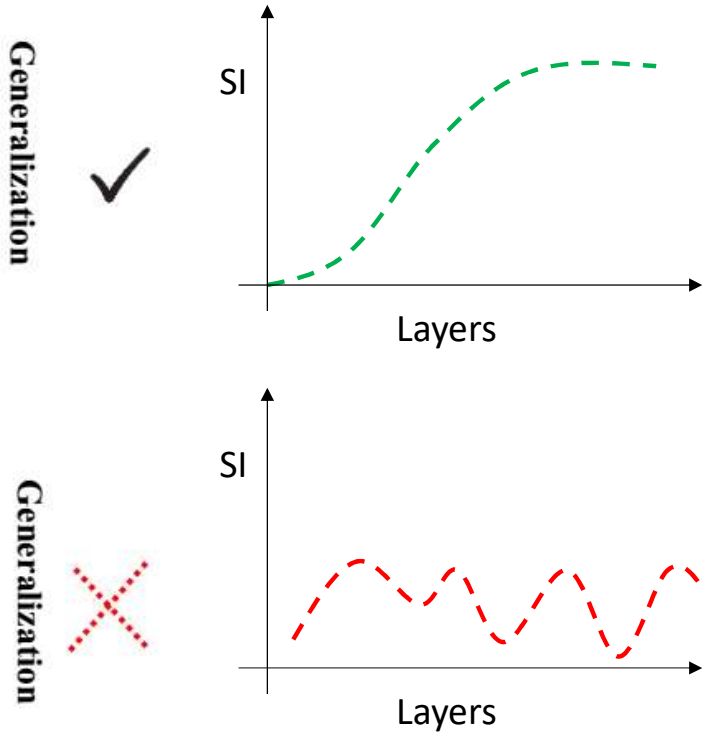
Some important notes

1. In fact, by decreasing the complexity of dataflow through “filter” layers, the SI(SmI) will increase.
2. The SI(SmI) may increase by different trends: different rises, different settling, with smooth or oscillatory changes.
3. Increasing SI(SmI) by Fully Connected (FC) layers is not desired. FC layers due to its redundancies make overfitting and avoid generalization.

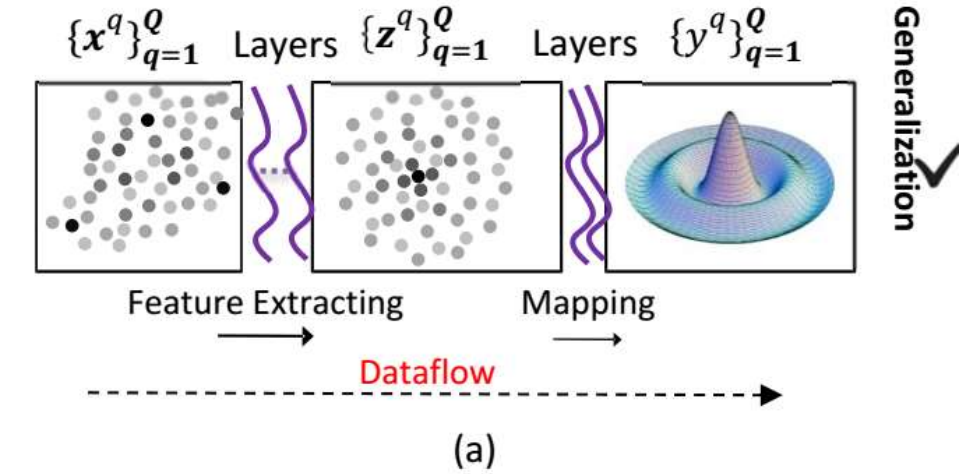


Generalization and Separation index

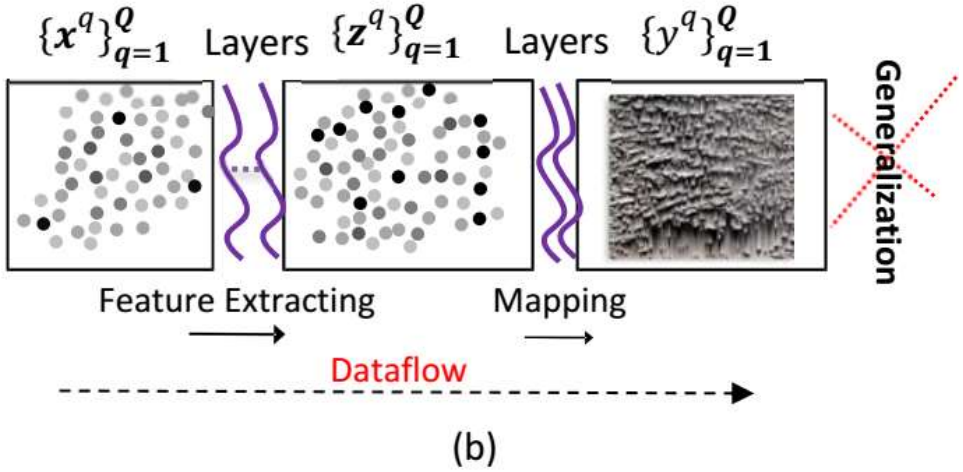




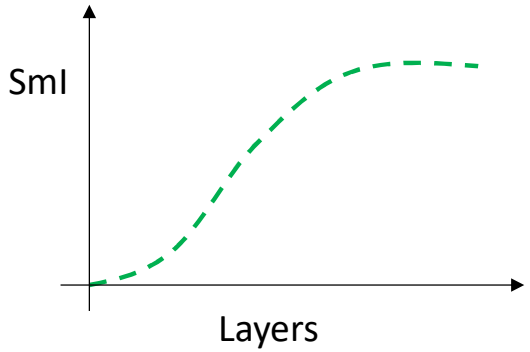
Generalization and Smoothness index

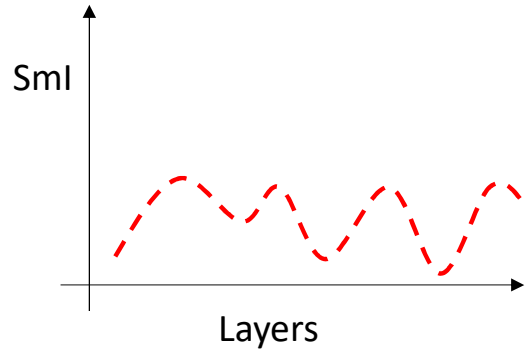


(a)



(b)





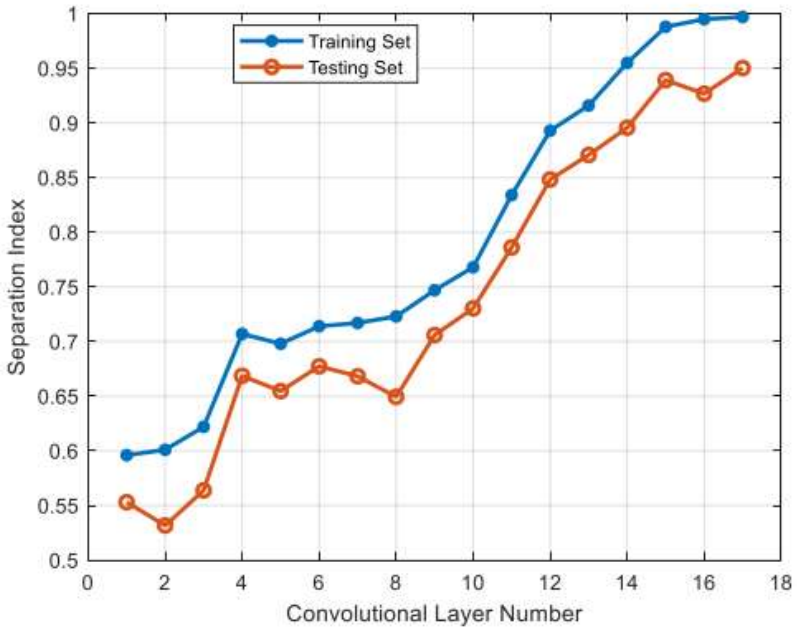
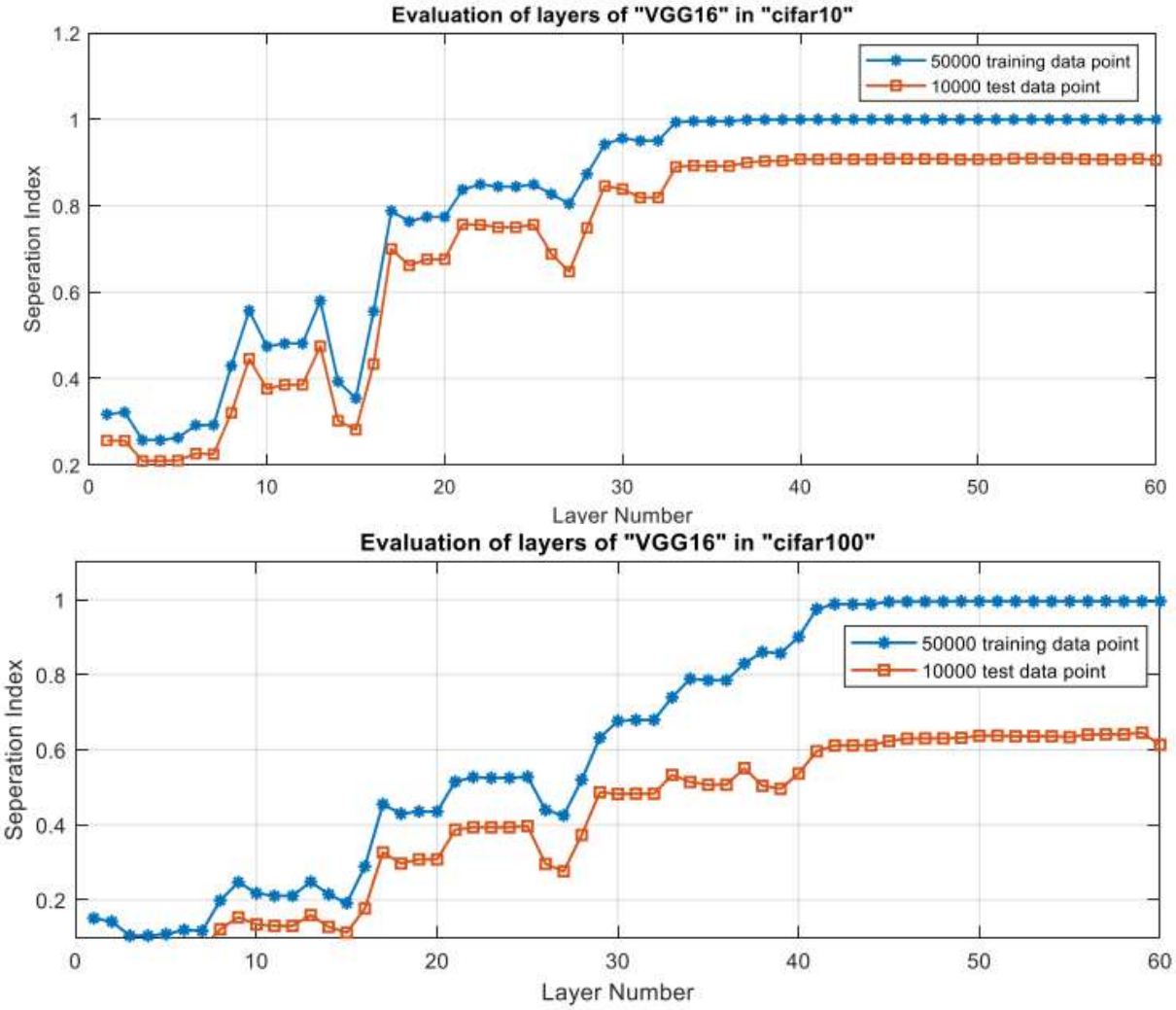


Fig. 10. Evaluation of dataflow through layers of the "Resnet18" network in the classification of Fashion-MNIST.



Correct Classification rate and SI

To compare correct classification rate and SI:

1. After each convolution layer of a pertained network, two dense layers are added in order to predict the true labels. After two dense layers, a batch normalization layer is utilized and after them, a softmax layer is applied.
2. Considering the sum of squared error as the loss function, the "Adam" optimizer with more than 100 epochs has been utilized.
3. This process is performed on the "cifar10" dataset on pre-trained "VGG-16" and "Fashion-MNIST" dataset on trained "Resnet18" separately.

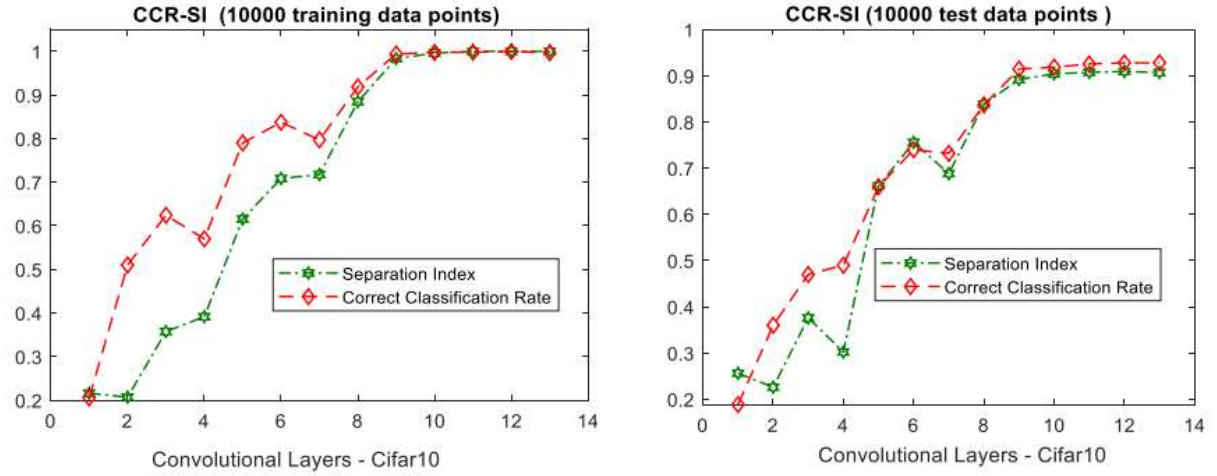


Fig. 11. The plots of separation index and correct classification rates at convolution layers in a pertained "vgg-16" network utilized for classification of "cifar10" for both training and test sets.

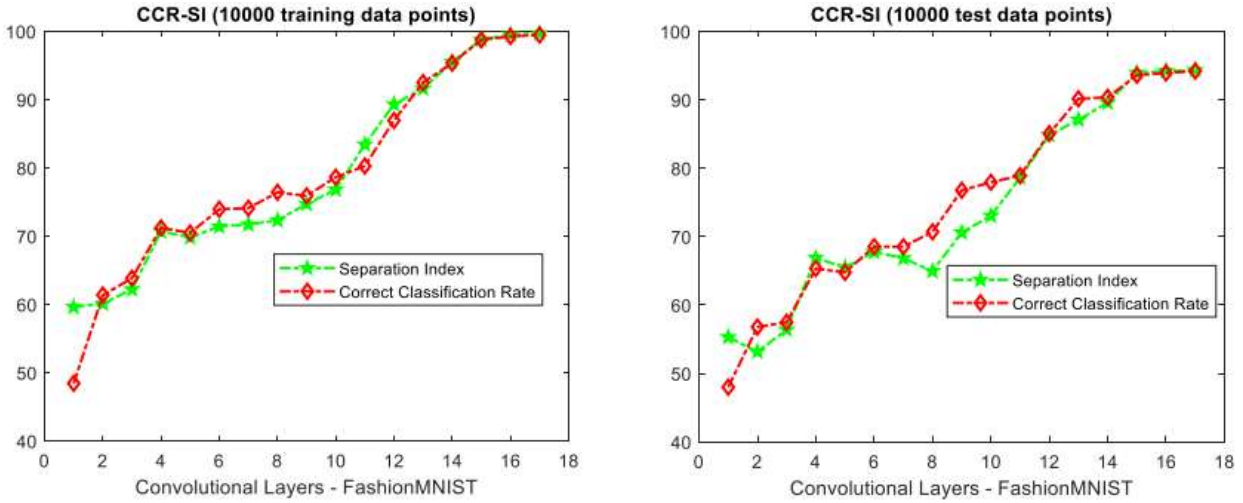


Fig. 12. The plots of separation index and correct classification rates at convolution layers in a trained "Resnet18" network utilized for classification of "Fashion-MNIST" for both training and test sets.

Pre-train Model ranking

- Assume there are M different pre-trained models which are trained by M different source data.
- Assume for j=1,2,…,M, $there\ are\ \ L_j$ layers before fully connected layers .
- Now, we want rank n_{model} pre-trained models to be used in transfer learning for a target data:

$Data$=$\{(x_i, y_i)\}_{i=1}^{m}$, y_i for classification problems denotes the same label l_i

Algortihm2: (To suggest a pre-trained model in transfer learning)

1. Apply the target data to jth pre-trained model and provide following dataflow at the last layer before fully connected layers:

$$Data^{j,L_j}=\left\{\left(x_i^{j,L_j}, y_i\right)\right\}_{i=1}^{m}$$

2. For jth pre-trained model select the best subset of x_i^{j,L_j} as $x_i^{j^*,L_j}$ which maximizes $SI(SmI)$

$$Data^{j^*,L_j}=\left\{\left(x_i^{j^*,L_j}, y_i\right)\right\}_{i=1}^{m} \qquad x_i^{j^*,L_j} \subseteq x_i^{j,L_j}$$

3. Now rank the pre-trained models as best candidates for the target data in transfer learning as follows:

$$Best\_Models = \{j_1 \dots, j_M\} \quad \text{where} \quad SI\left(Data^{j_1^*,L_{j_1}}\right) > SI\left(Data^{j_2^*,L_{j_2}}\right) > \cdots > SI\left(Data^{j_M^*,L_{j_M}}\right)$$

Model Confidence and Guarantee (SI, SmI)

Assumptions

1. There is a training dataset $Data = \{(x_i, l_i)\}_{i=1}^{m}$ ($Data = \{(x_i, y_i)\}_{i=1}^{m}$) for a classification(regression) problem.
2. The $SI(Data)$ ($SmI\ (Data)$) is computed for the dataset.
3. There is a test dataset which is homogenous with the training dataset.
4. Based on the Nearest Neighbor (NN) model, we want to predict the target $l\ (\boldsymbol{y})$ of a new test example $\boldsymbol{x}$, as $\hat{l}(\hat{\boldsymbol{y}})$.
5. It is assumed that $x_{i_j^*}$ denotes the jth nearest neighbor of input data to x.
6. There is an uncertainty in measuring x . It is assumed that x has maximum measuring error (distance) γ to its true value x_{true}: $\|x_{true} - x\| < \gamma$.
7. It is aimed to know that the confidence of the prediction by the NN model.
8. It is aimed to know if there is a guarantee to predict true label in classification problem or to have a limited output error in a regression problem.

Model Confidence and Guarantee by SI (without training data)

Assume dw_{max} denotes the maximum distance of nearest neighbor exampled with the the same label and db_{min} denotes the minimum inter distance between examples with different labels .

$Conf(\hat{l}, l_{i_1^*})$ denotes the confidence for prediction of $\hat{l} = l_{i_1^*}$(by the NN model).

Guar ($\hat{l}, l_{i_1^*}$) denotes the guarantee that prediction of $\hat{l} = l_{i_1^*}$(by the NN model) is true.

if Guar ($\hat{l}, l_{i_1^*}$) =1 the guarantee exists and otherwise it does not exist.

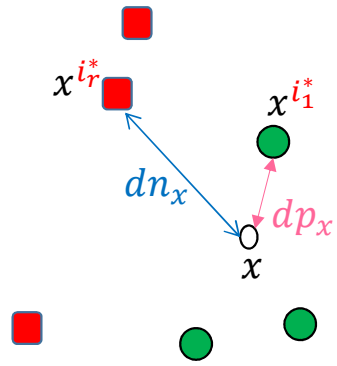
- $Conf(\hat{l}, l_{i_1^*})$=$SI(Data)$
- $Guar(\hat{l}, l_{i_1^*}) = \delta(SI, 1) * sign(1 - \frac{dw_{max}+\gamma}{db_{min}-\gamma})$

γ: the maximum uncertainty in measuring x :$\|x_{true} - x\| < \gamma$

❖ If $SI(Data) = 1$ and dw_{max}<db_{min}, the true prediction for $x\ with\ NN\ model$ and for $\gamma < 0.5(db_{min} - dw_{max})$ is guaranteed.

Model Confidence and Guarantee by SI (with training data)

$$dp_x = \left\| x - x_{i_1^*} \right\|$$
$$dn_x = \left\| x - x_{i_r^*} \right\| \quad r = \min_{j=1,..,m} j \text{ subjec to } \delta(l_{i_j^*}, l_{i_1^*}) = 0$$



- $Conf(\hat{l}, l_{*_1}) = SI$
- $Guar(\hat{l}, l_{*_1}) = \delta(SI, 1) * sign(1 - \frac{dp_x + \gamma}{dn_x - \gamma})$

❖ If $SI(Data) = 1$ the true prediction for x $with$ NN $model$ and $\gamma < 0.5(dn_x - dp_x)$ is guaranteed.

- About those cases that we have $SI^r(Data) = 1$ for $r \gg 1$ the guarantee is satisfied for much higher γ than the cases which we have $SI^1(Data) = 1$.