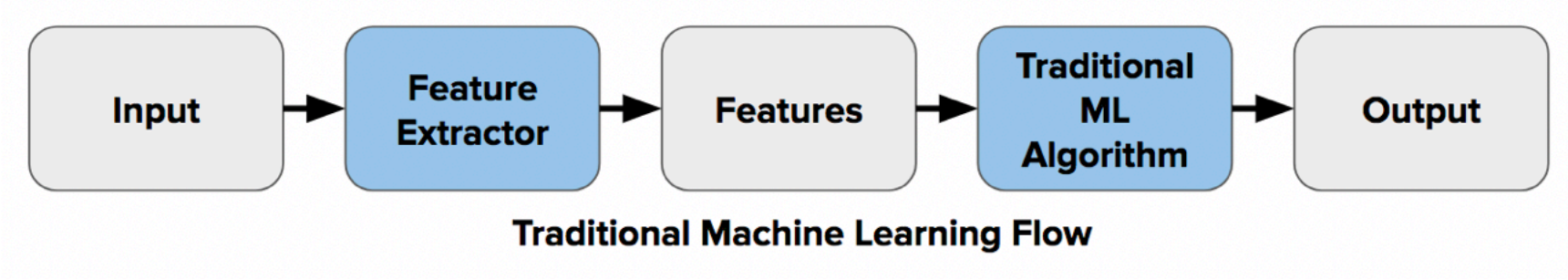
Introduction to Convolutional Neural Networks

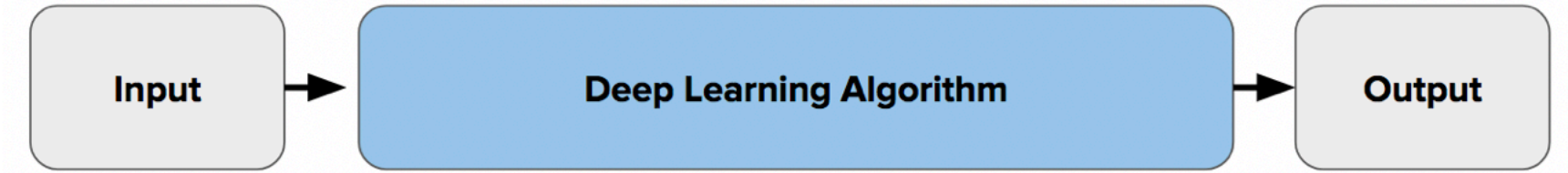
Lê Anh Cường

TDTU

Traditional Statistical Machine Learning vs Deep Learning

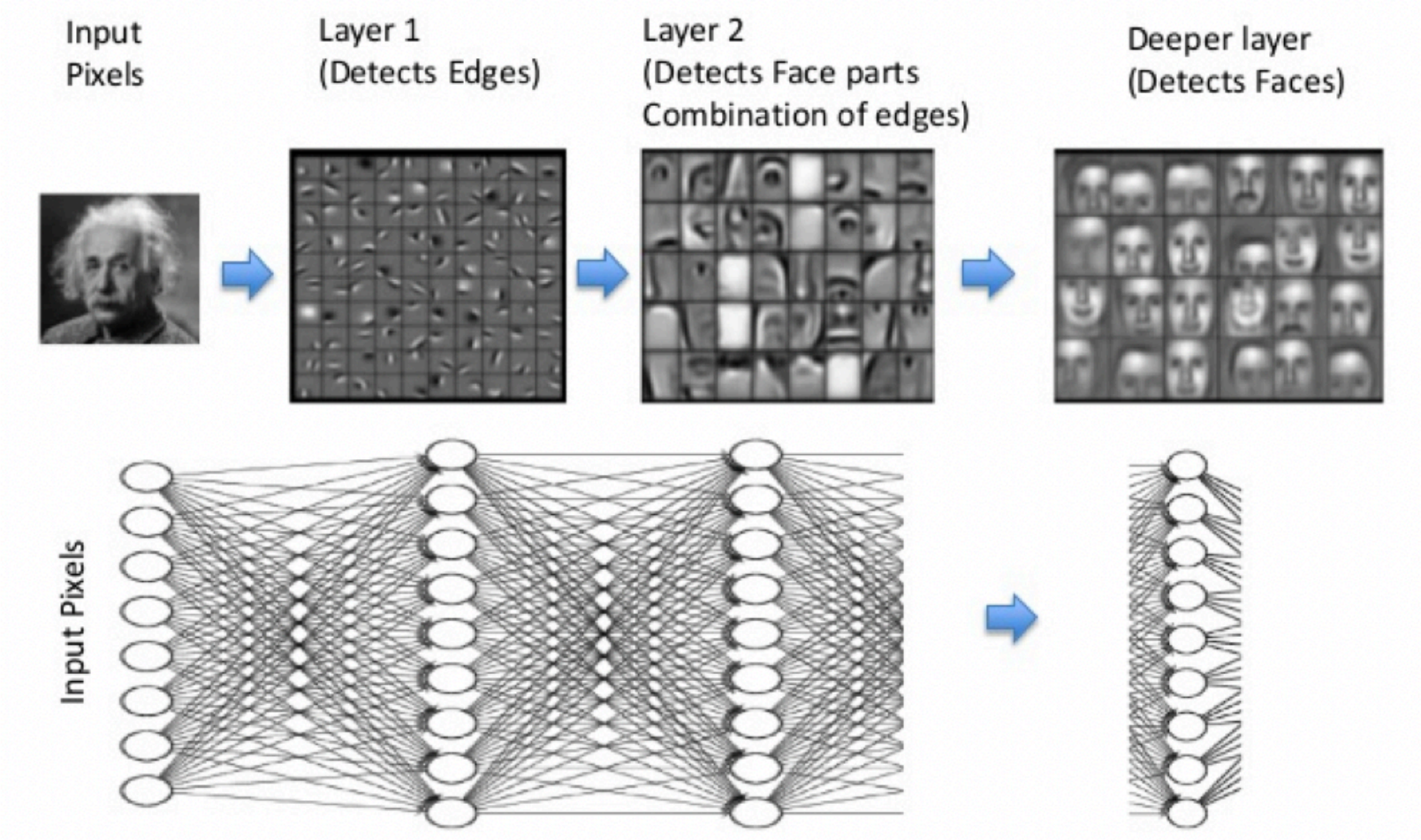


Traditional Machine Learning Flow

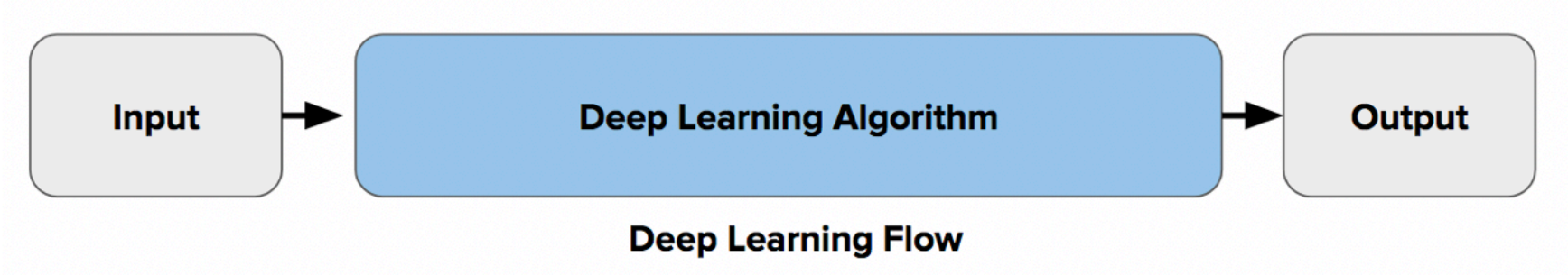


Deep Learning Flow

DL as Representation Learning

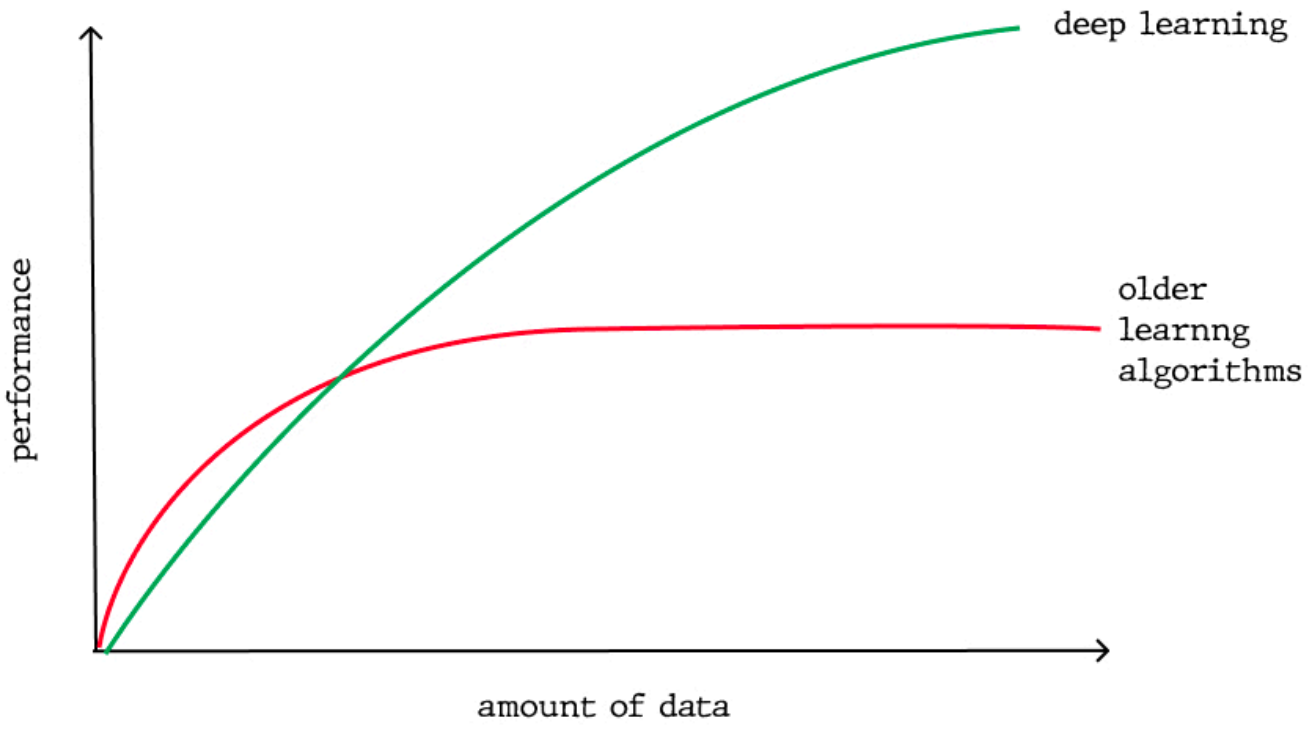


End-to-End Model

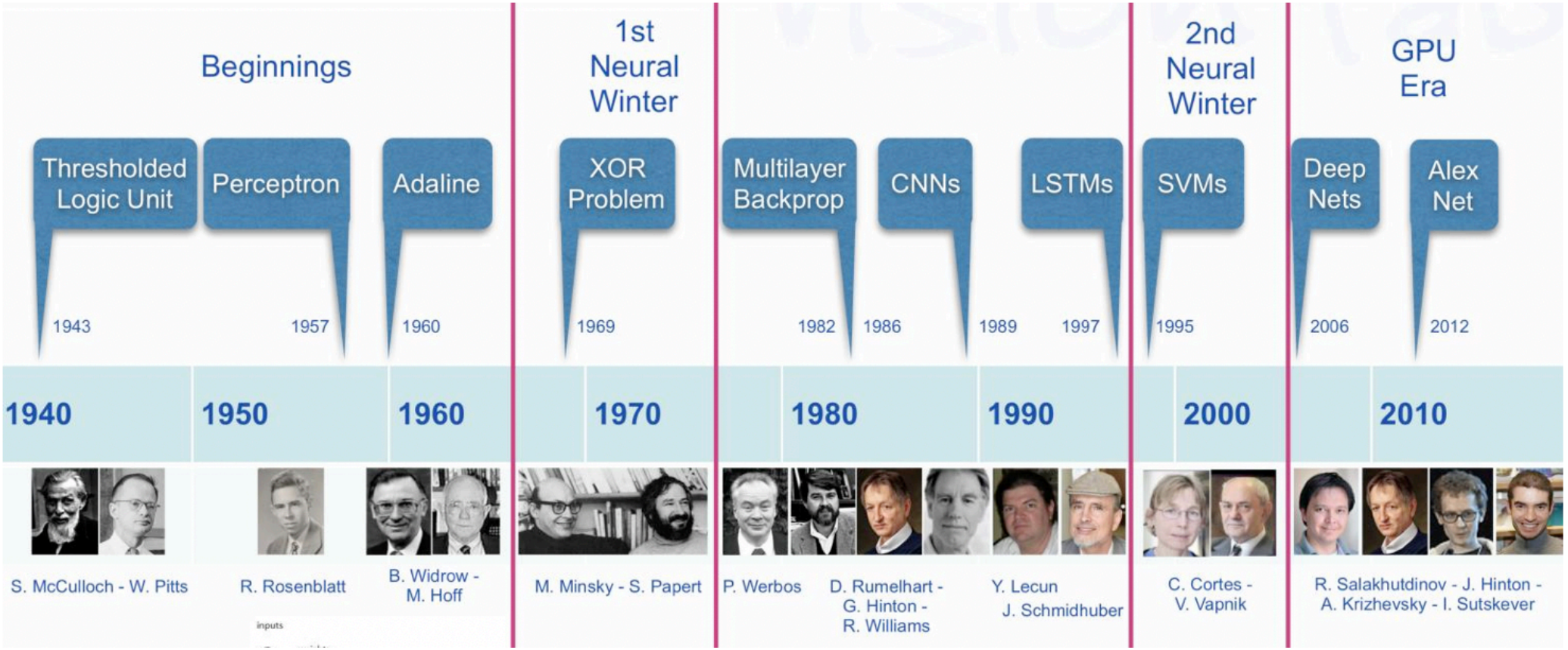


Deep Learning Flow

ML vs DL in Performance



History of Deep Learning



Recent DL models

- Generative Adversarial Network (2014)
- Residual Neural Network (2015)
- Transformer (2017)

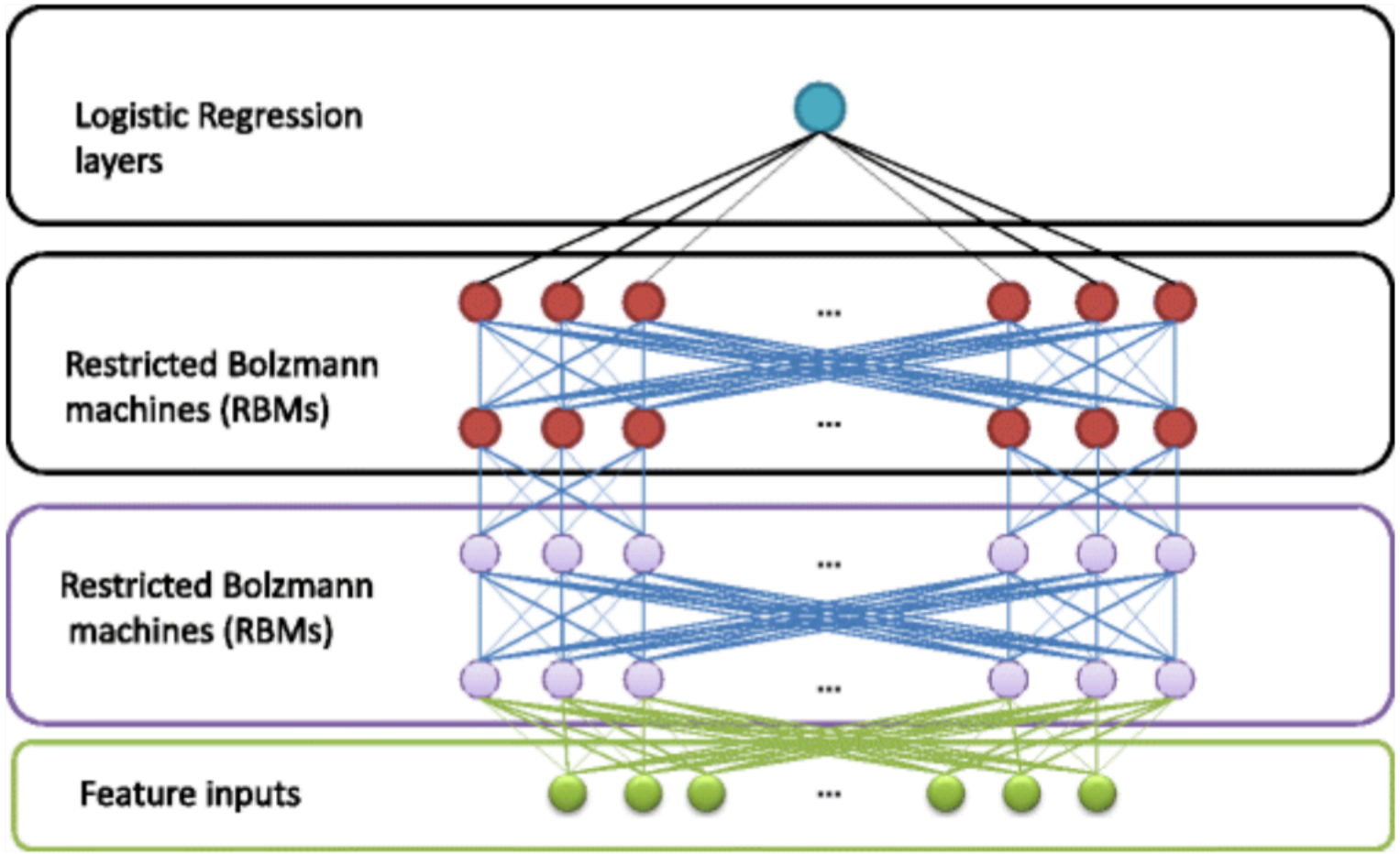
When the term Deep Learning?

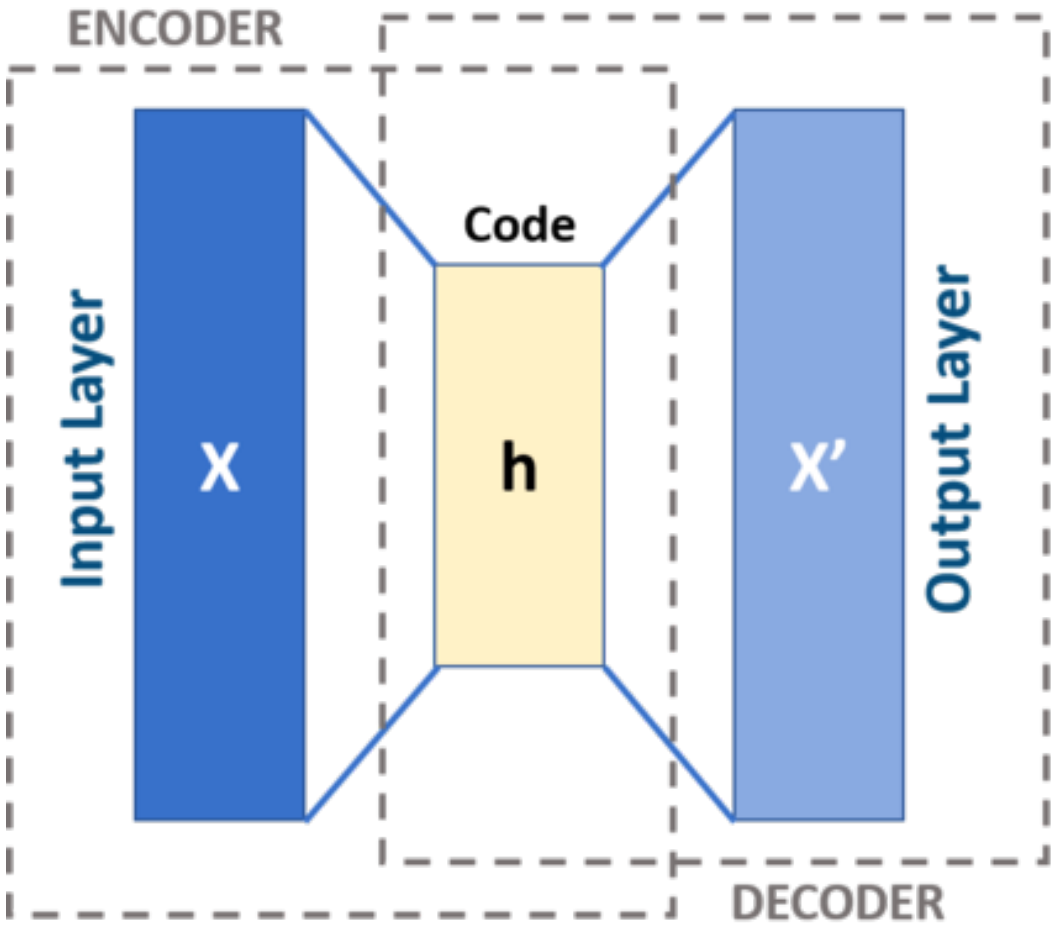
Geoffrey Hinton is a pioneer in the field of artificial neural networks and co-published the first paper on the backpropagation algorithm for training multilayer perceptron networks.

He may have started the introduction of the phrasing "*deep*" to describe the development of large artificial neural networks.

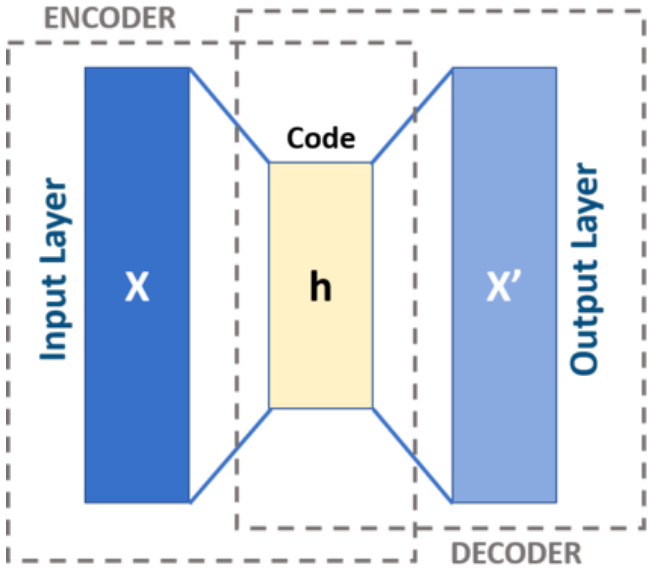
He co-authored a paper in 2006 titled "A Fast Learning Algorithm for Deep Belief Nets" in which they describe an approach to training "deep" (as in a many layered network) of restricted Boltzmann machines.

Deep Belief Network





An autoencoder is a type of artificial neural network used to learn efficient data codings in an unsupervised manner.[1] The aim of an autoencoder is to learn a representation (encoding) for a set of data, typically for dimensionality reduction



An autoencoder is a type of artificial neural network used to learn efficient data codings in an unsupervised manner.[1] The aim of an autoencoder is to learn
a representation (encoding) for a set of data, typically for dimensionality reduction

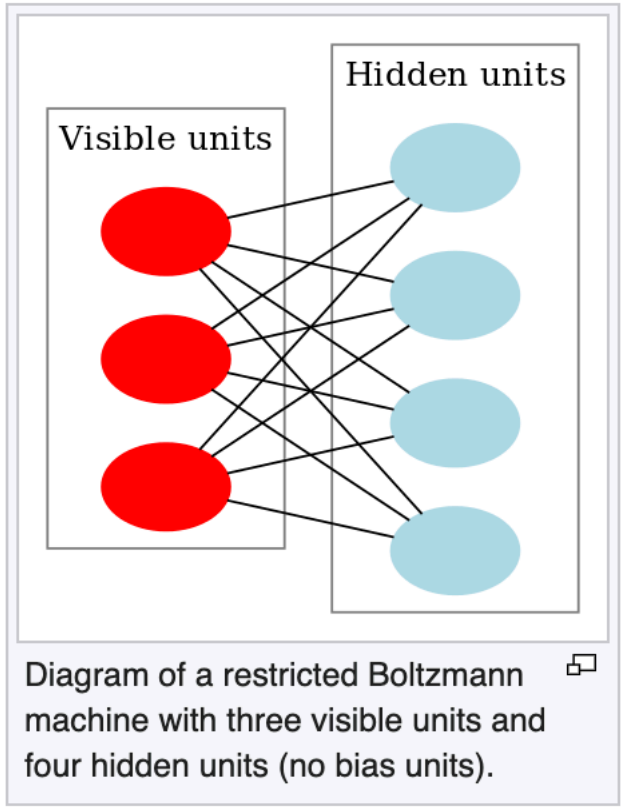


Diagram of a restricted Boltzmann machine with three visible units and four hidden units (no bias units).

A **restricted Boltzmann machine** (**RBM**) is a generative stochastic artificial neural network that can learn a probability distribution over its set of inputs.

Recognition of DL

- Deep: i.e. many layers/levels of data representation
- Big Data (large enough): for learning good features
- High Performance Computing: for doing with complex networks and big data

Convolutional Neural Network

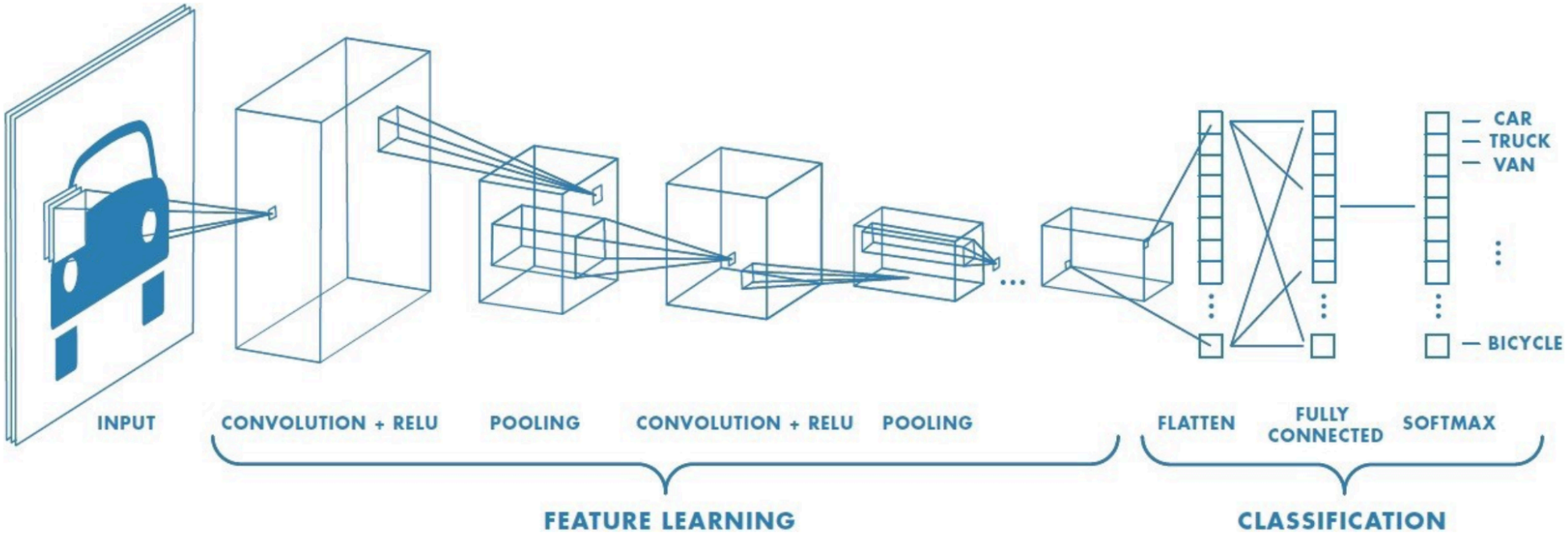
Outline

- What is CNN?
- Why CNN?
- CNN Explanation
- Implementation

ConvNet History

- Convolutional neural networks, also called ConvNets, were first introduced in the 1980s by Yann LeCun for building an image recognition neural network.
- LeNet-5, a pioneering 7-level convolutional network by LeCun et al. in 1998, that classifies digits, was applied by several banks to recognize hand-written numbers on checks.
- Although CNNs were invented in the 1980s, their breakthrough in the 2000s required fast implementations on graphics processing units (GPUs).
- A similar GPU-based CNN by Alex Krizhevsky et al. won the ImageNet Large Scale Visual Recognition Challenge 2012. A very deep CNN with over 100 layers by Microsoft won the ImageNet 2015 contest

A general architecture of Convolutional Neural Networks

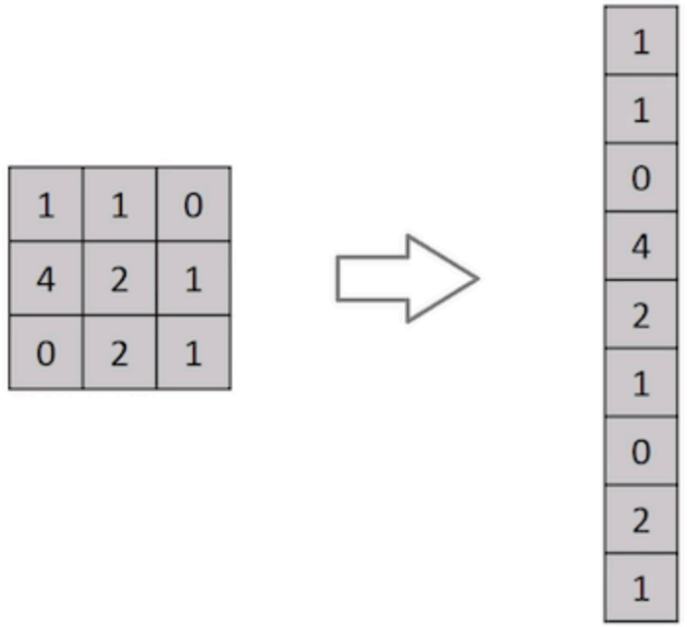


What is Convolutional Neural Network

- Convolutional Neural Networks (CNNs) are simply neural networks that use Convolution Operation which is a specialized kind of linear operation.
- CNNs are regularized versions of multilayer perceptrons.
- CNNs take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns.

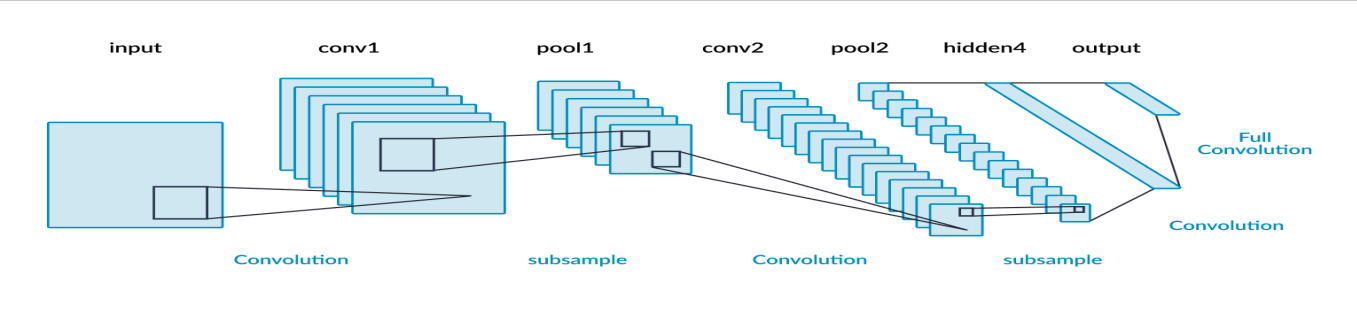
Why ConvNets over Feed-Forward Neural Nets?

- A ConvNet is able to **successfully capture the Spatial and Temporal dependencies.**
- The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters.
- The network can be trained to understand the sophistication of the image better.

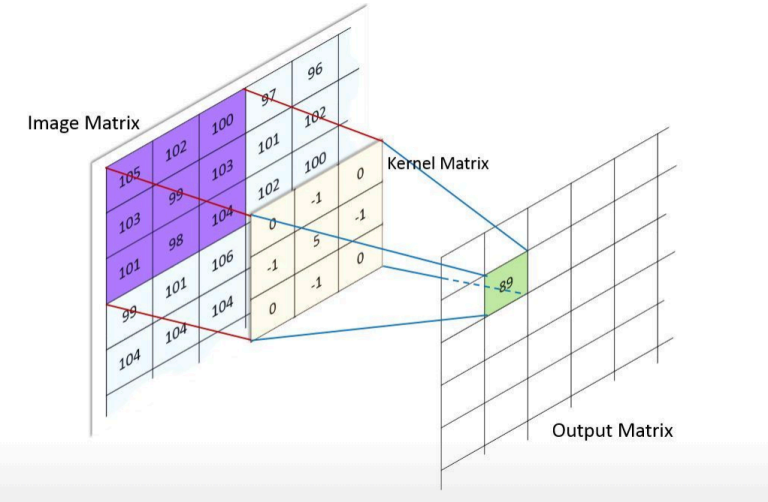


Flattening of a 3x3 image matrix into a 9x1 vector

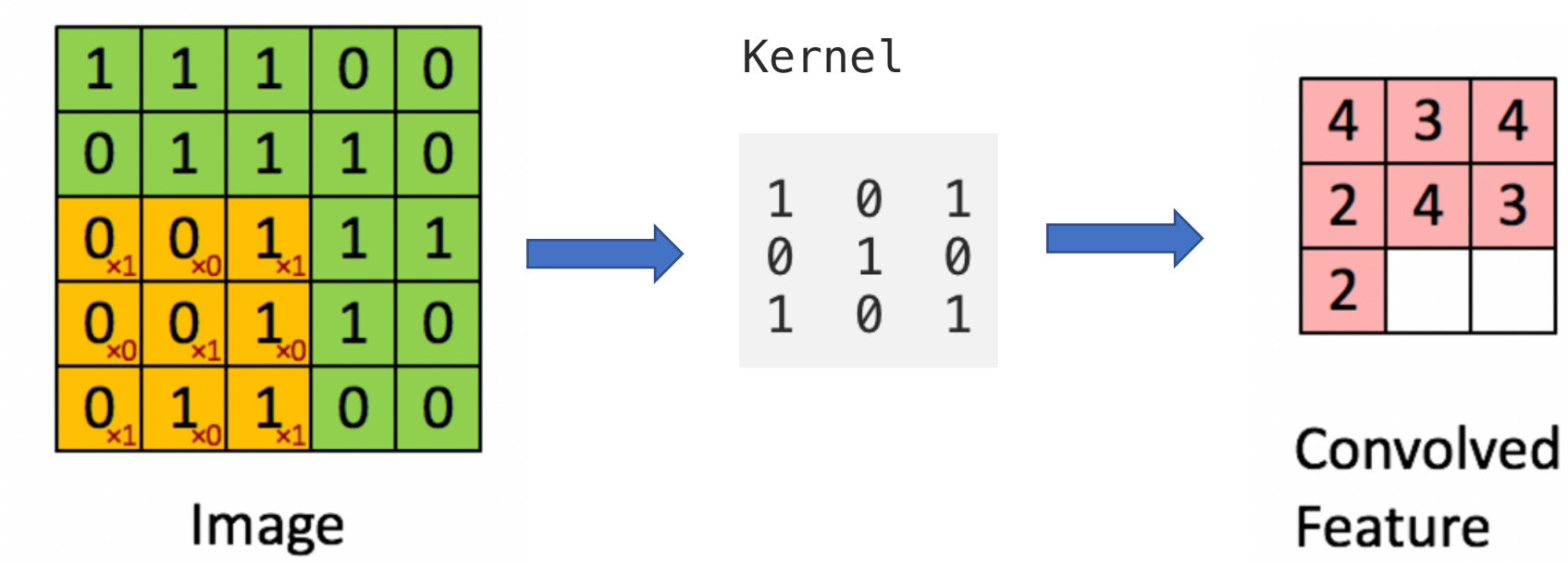
Convolution Layer — The Kernel



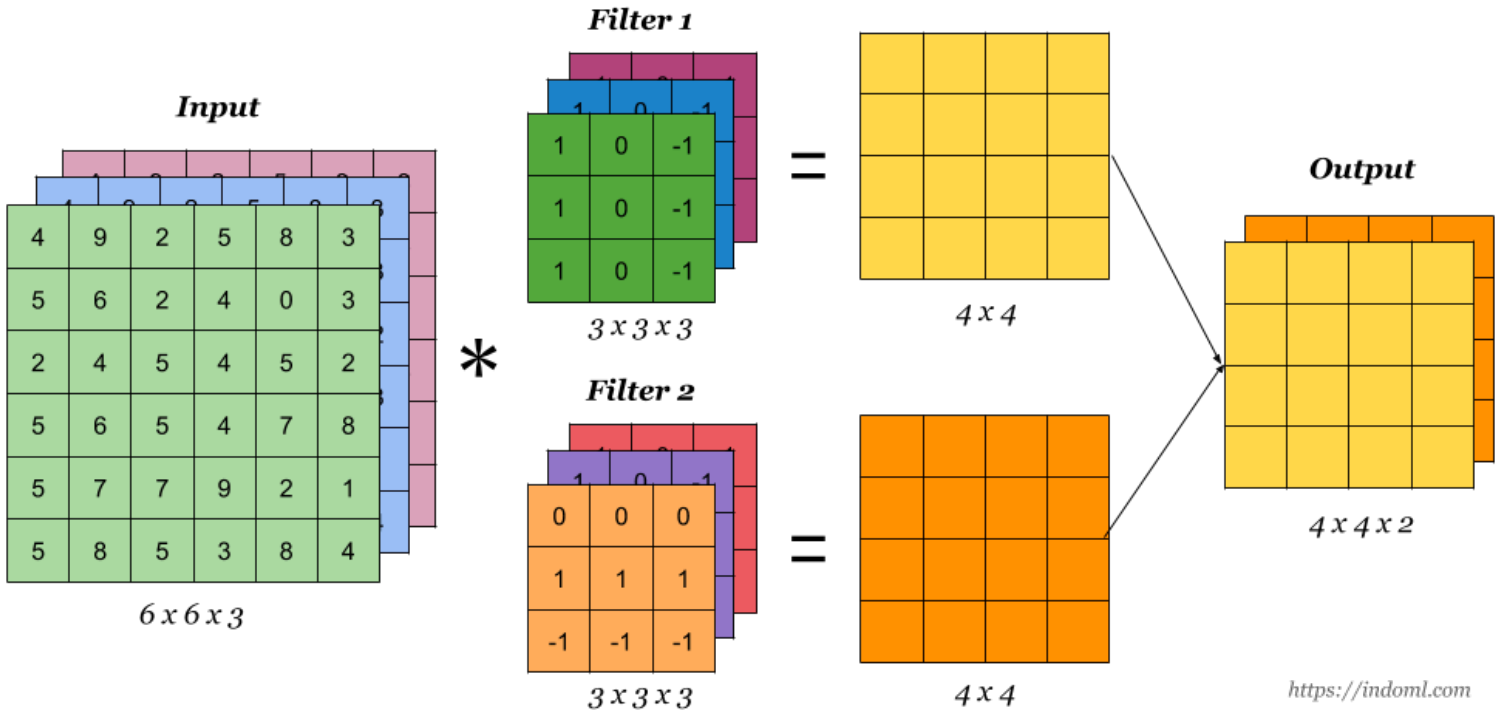
The objective of the Convolution Operation is to **extract the high-level features** such as edges, from the input image.



Convolution Layer — The Kernel

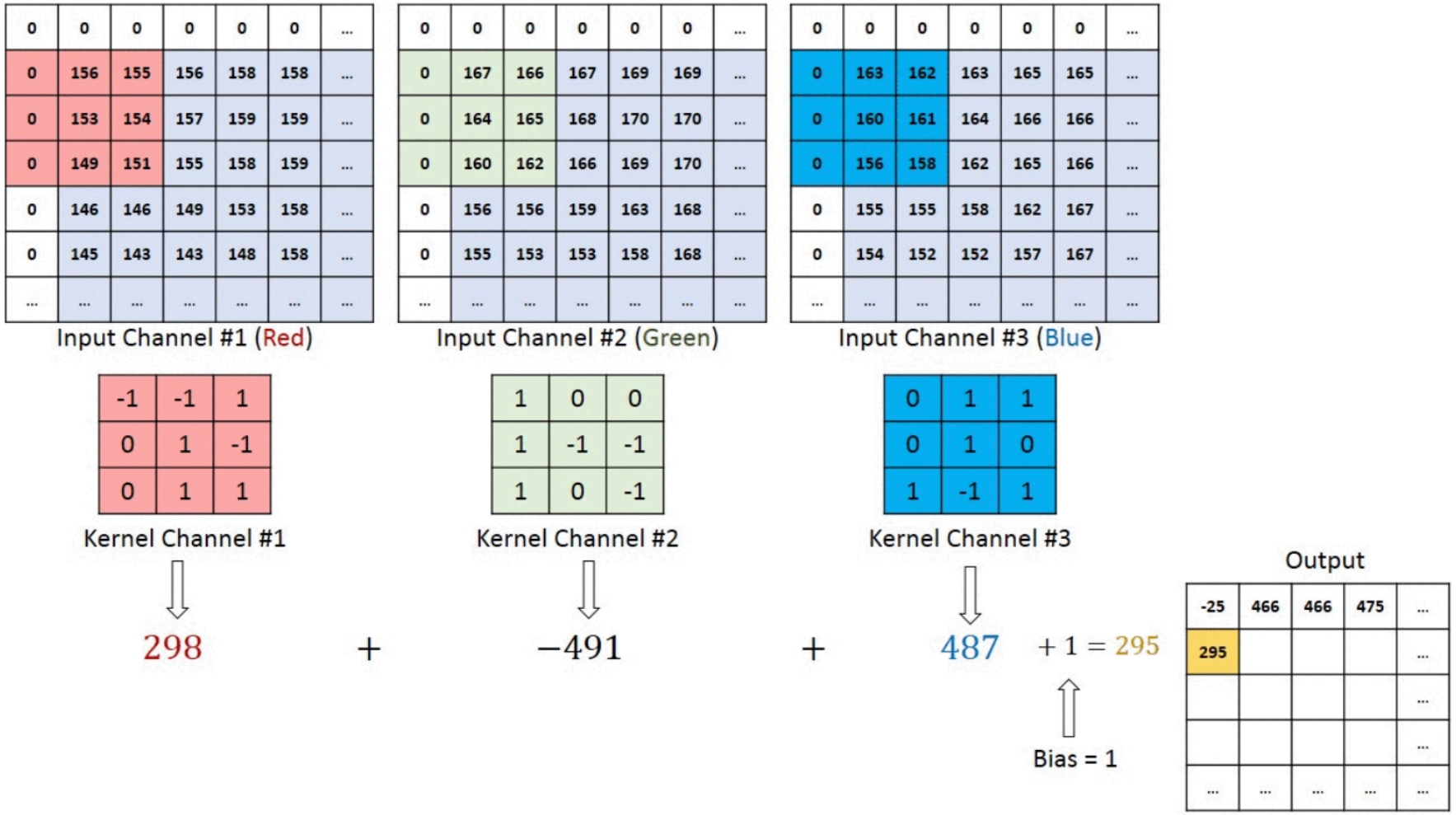


Multi-Channel Input and Filter



A common convolution layer actually consist of multiple such filters. *For the sake of simplicity in the discussion to follow, assume the presence of only one filter unless specified, since the same behavior is replicated across all the filters.*

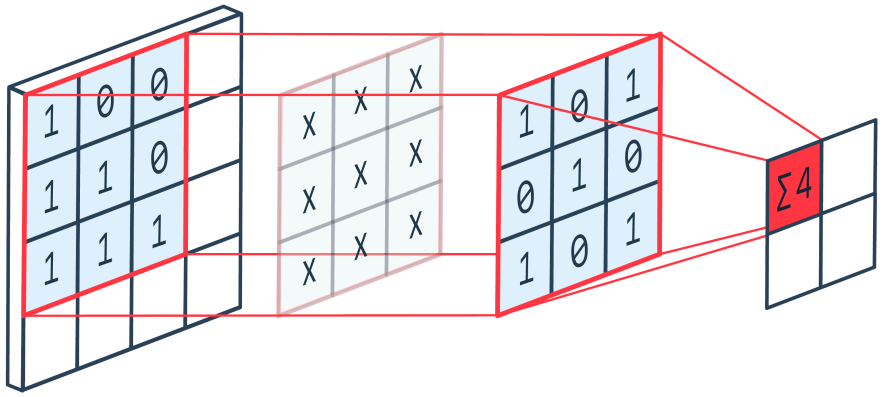
a *multi-input channel* convolution kernel

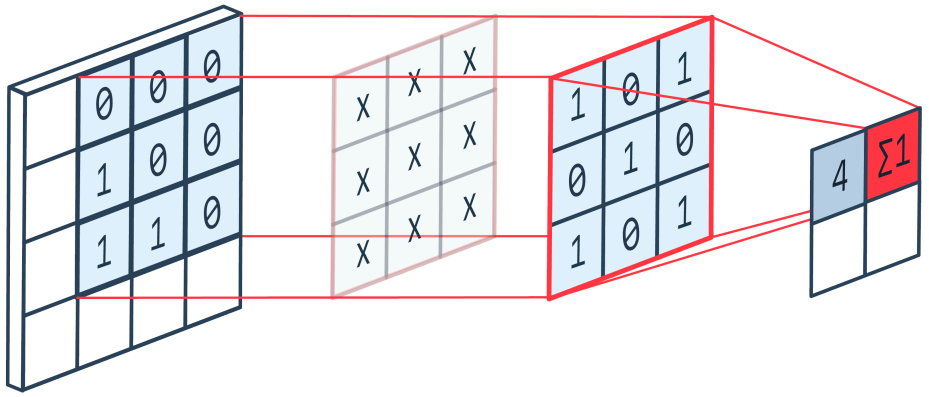


Convolution operation on a MxNx3 image matrix with a 3x3x3 Kernel

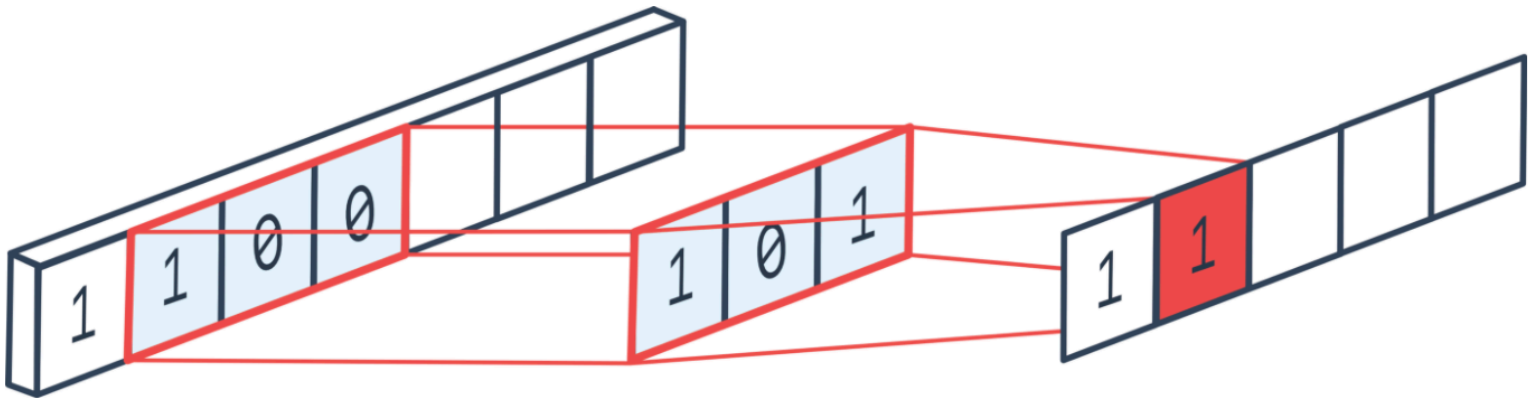
2D Convolutions

- The filter can move in 2 directions and thus the final output is 2D.
- 2D convolutions are the most common convolutions, and are heavily used in Computer Vision.

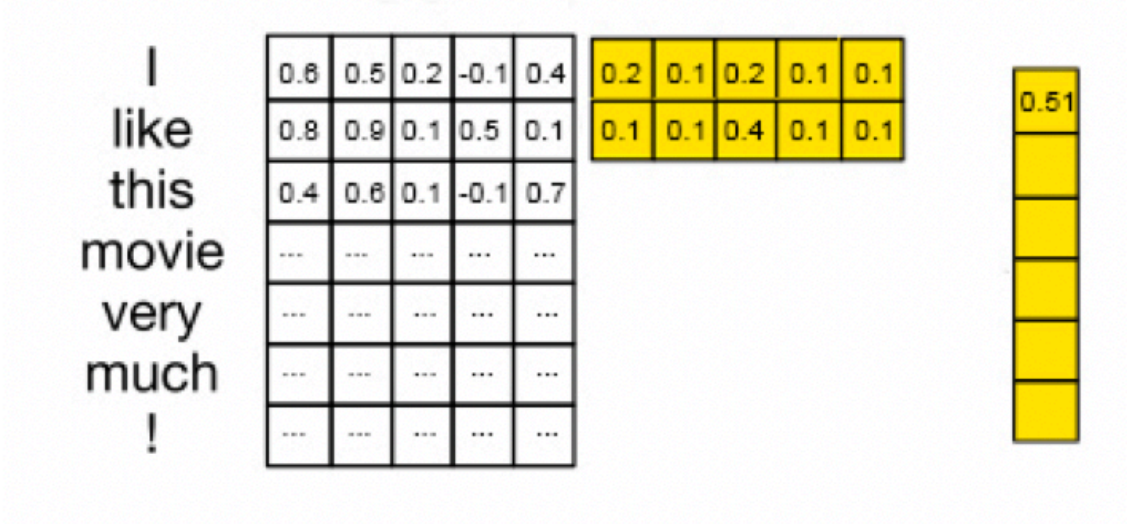


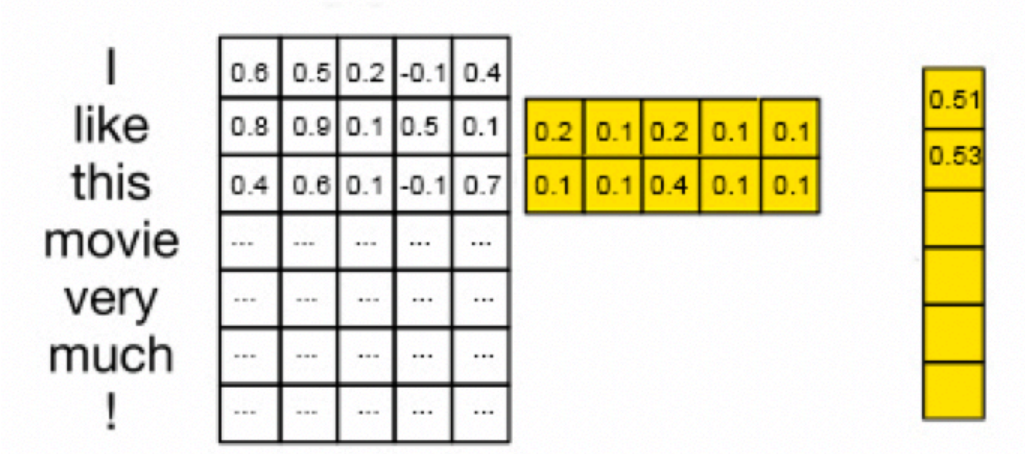


1D Convolutions

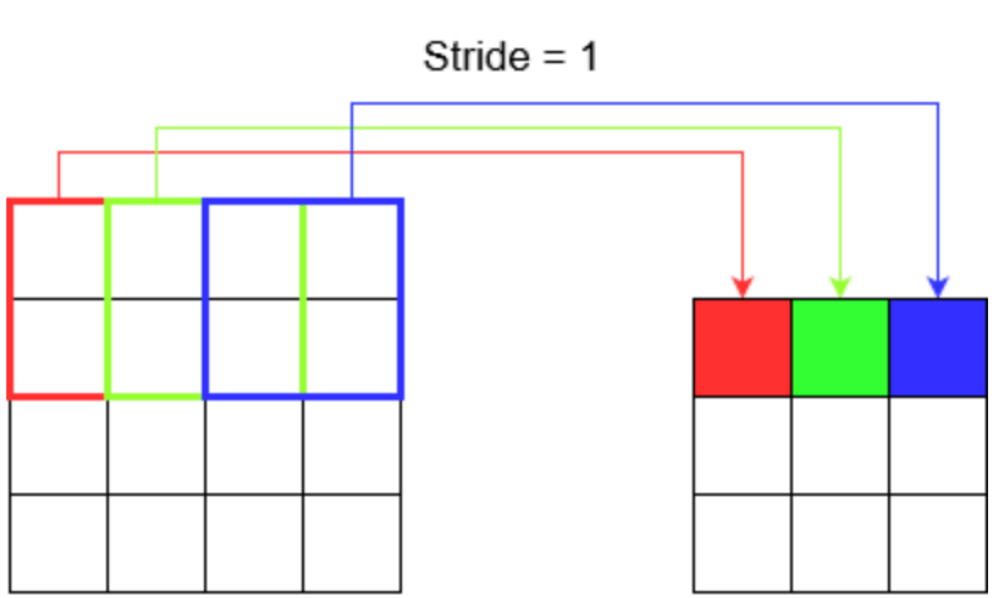


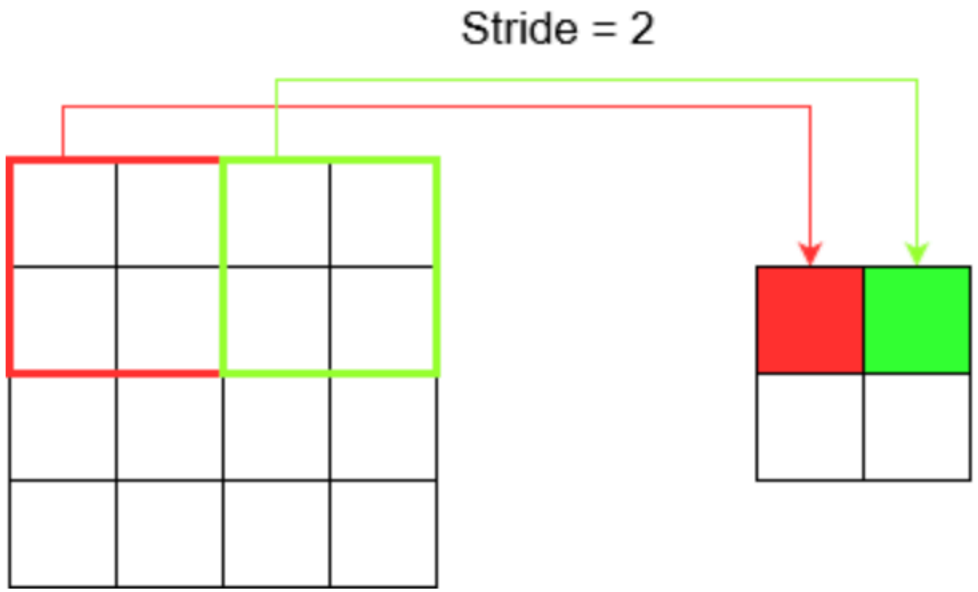
1D Convolution with 2D input



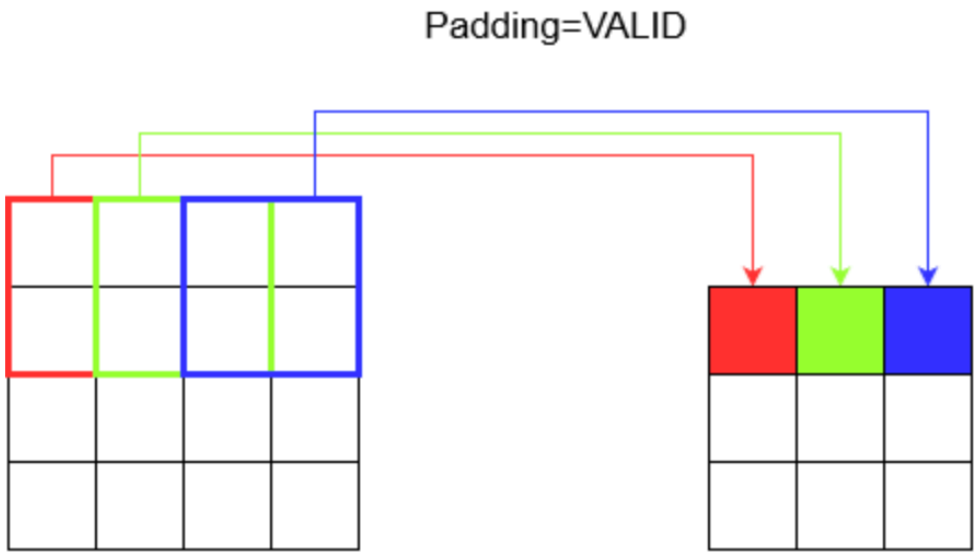


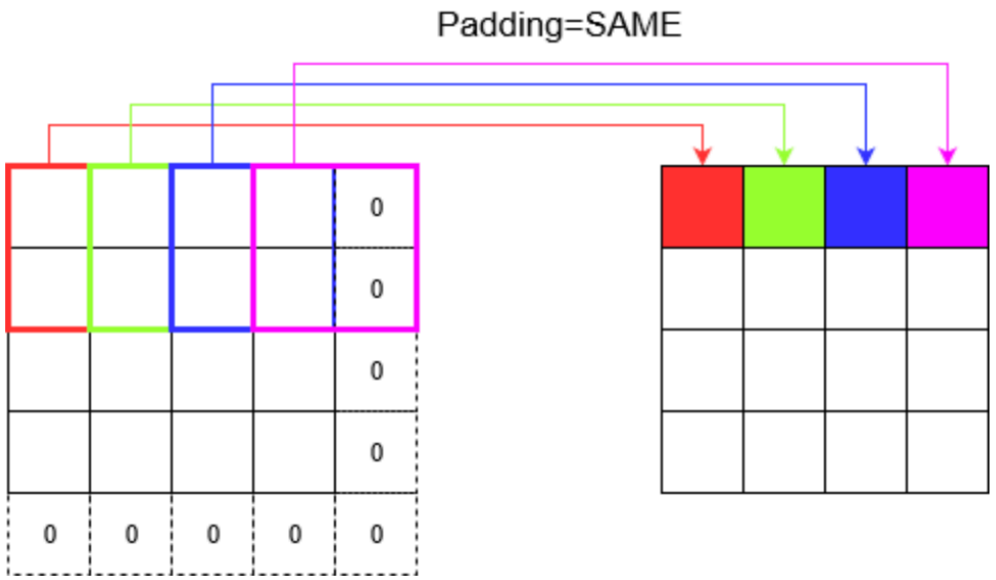
Stride





Padding

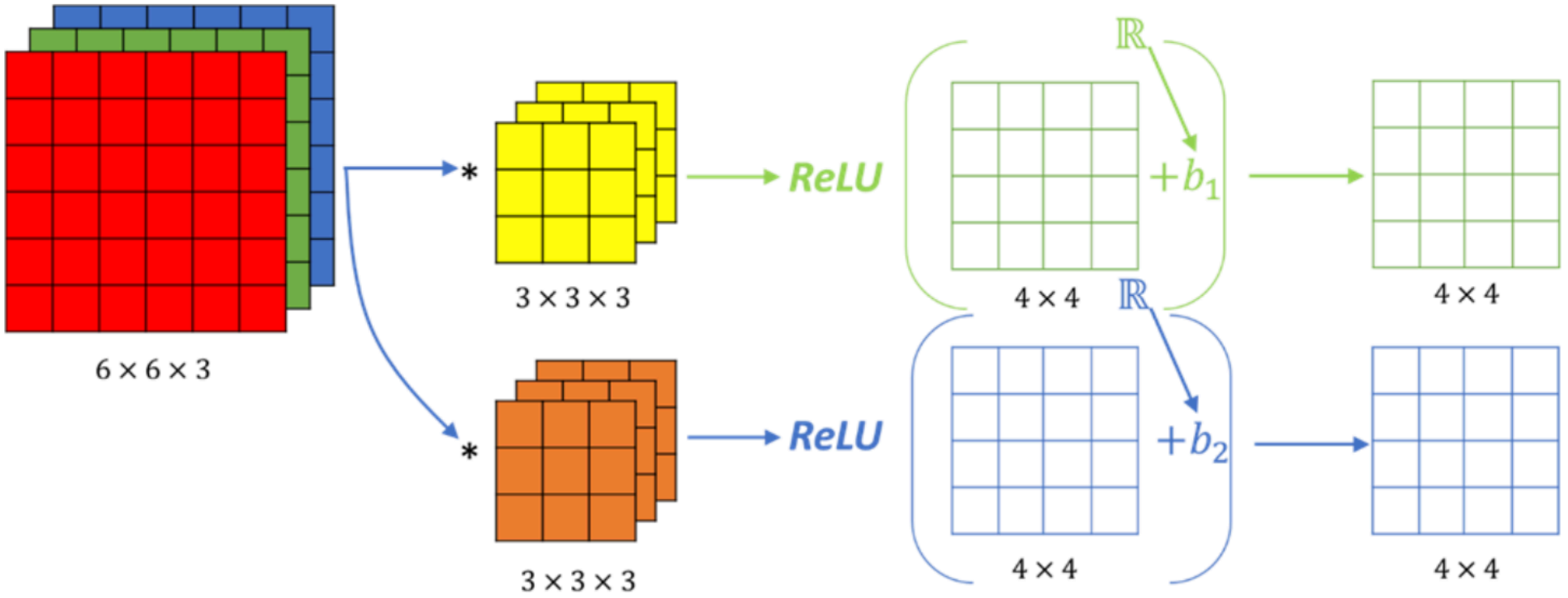




ReLU Operation

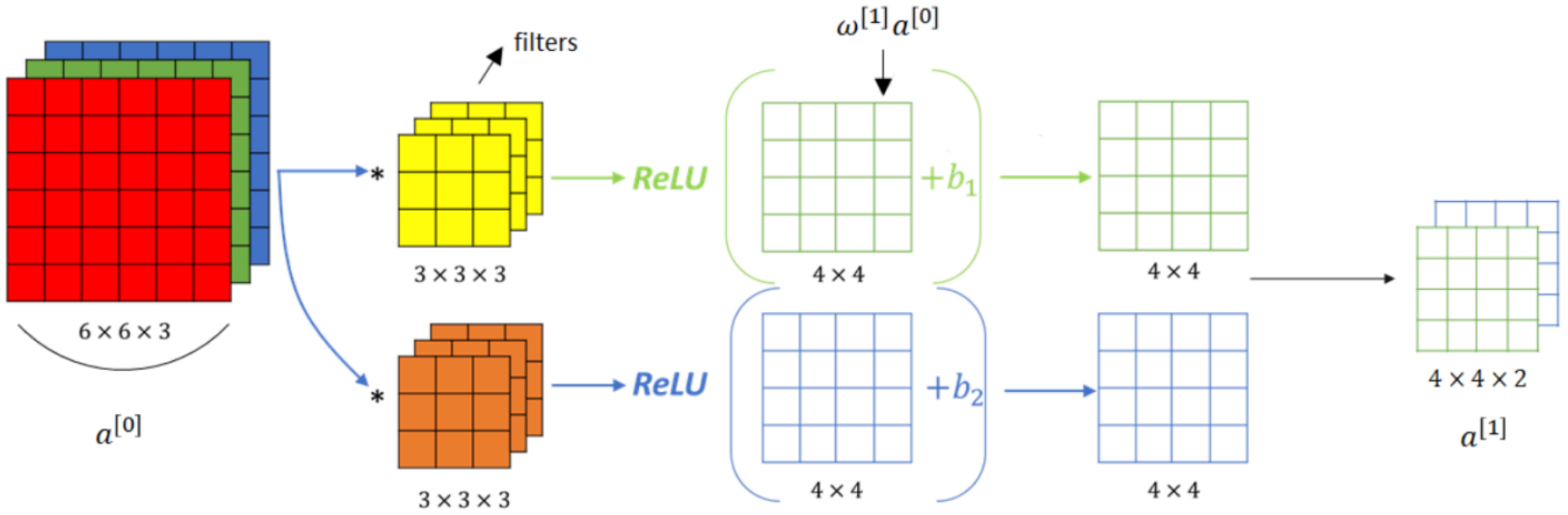
ReLU

$$\max(0, x)$$



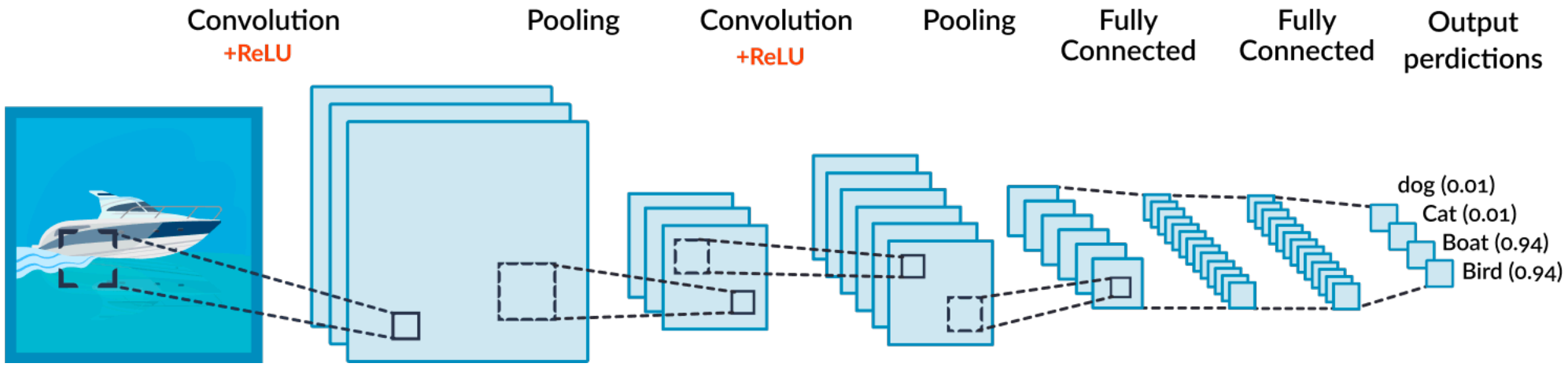
The result of convolution with two filters

ReLU



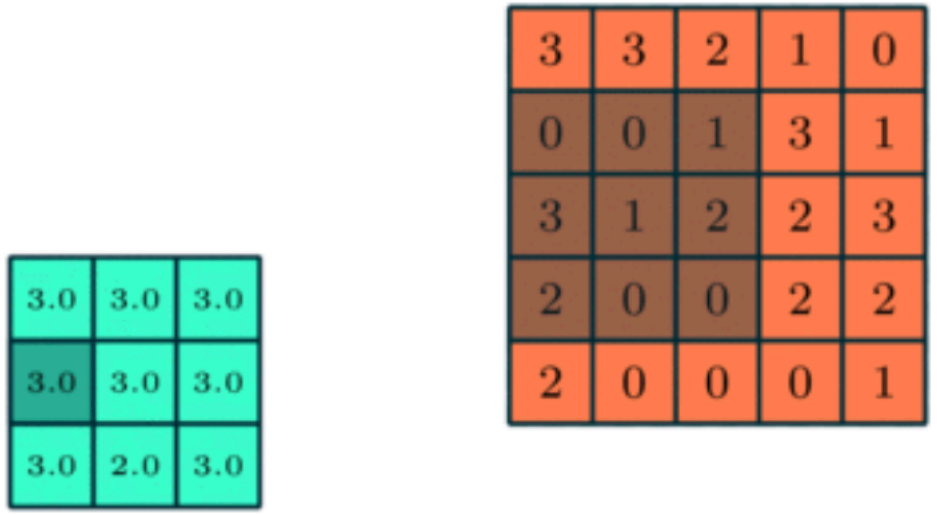
The result of a convolution of $6 \times 6 \times 3$ *with two* $3 \times 3 \times 3$ *is a volume of dimension* $4 \times 4 \times 2$

Pooling Layer



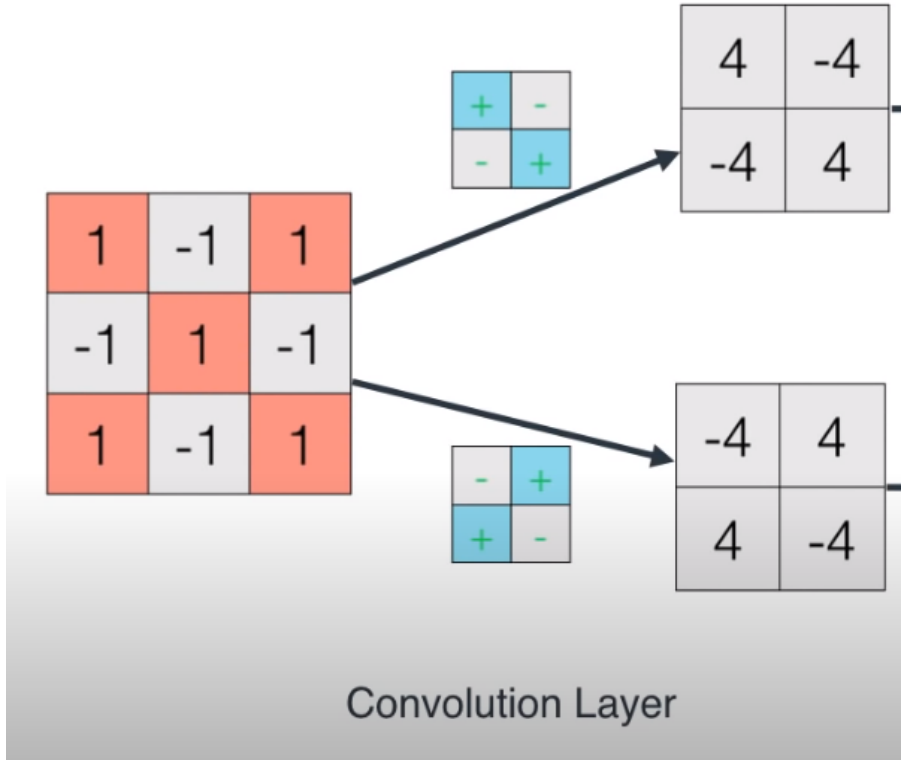
Pooling Layer

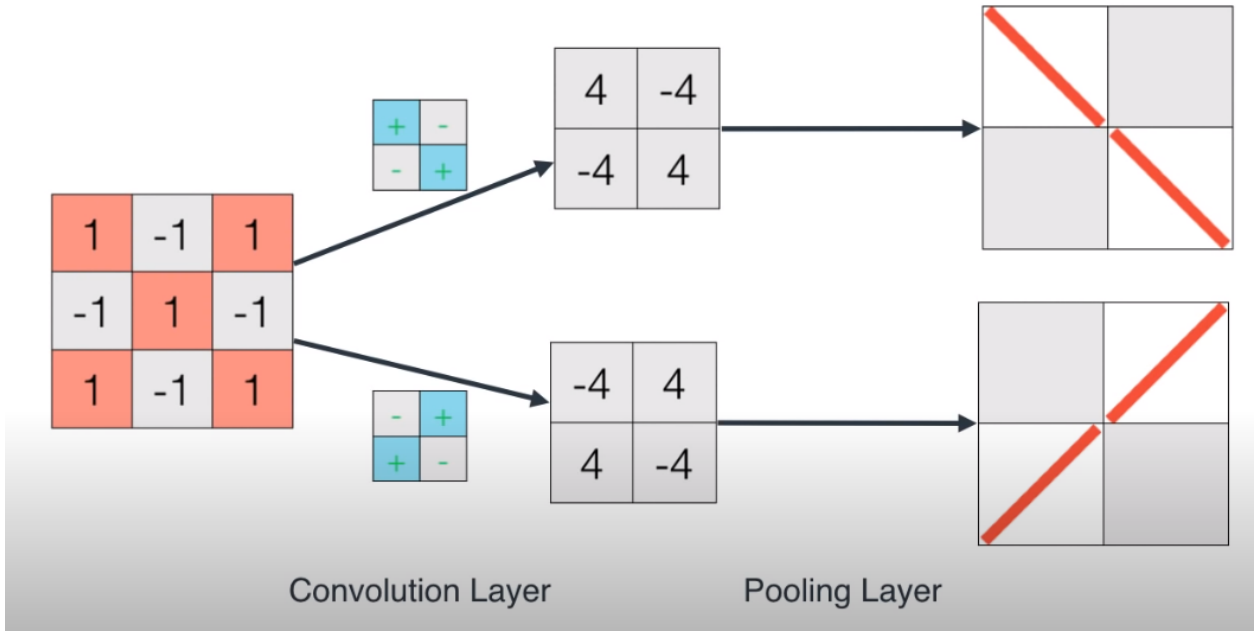
- The Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to **decrease the computational power required to process the data** through dimensionality reduction.
- Furthermore, it is useful for **extracting dominant features** .

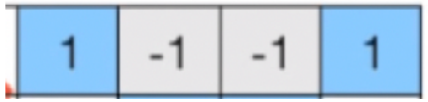


3x3 pooling over 5x5 convolved feature

Example for Convolutional Operations

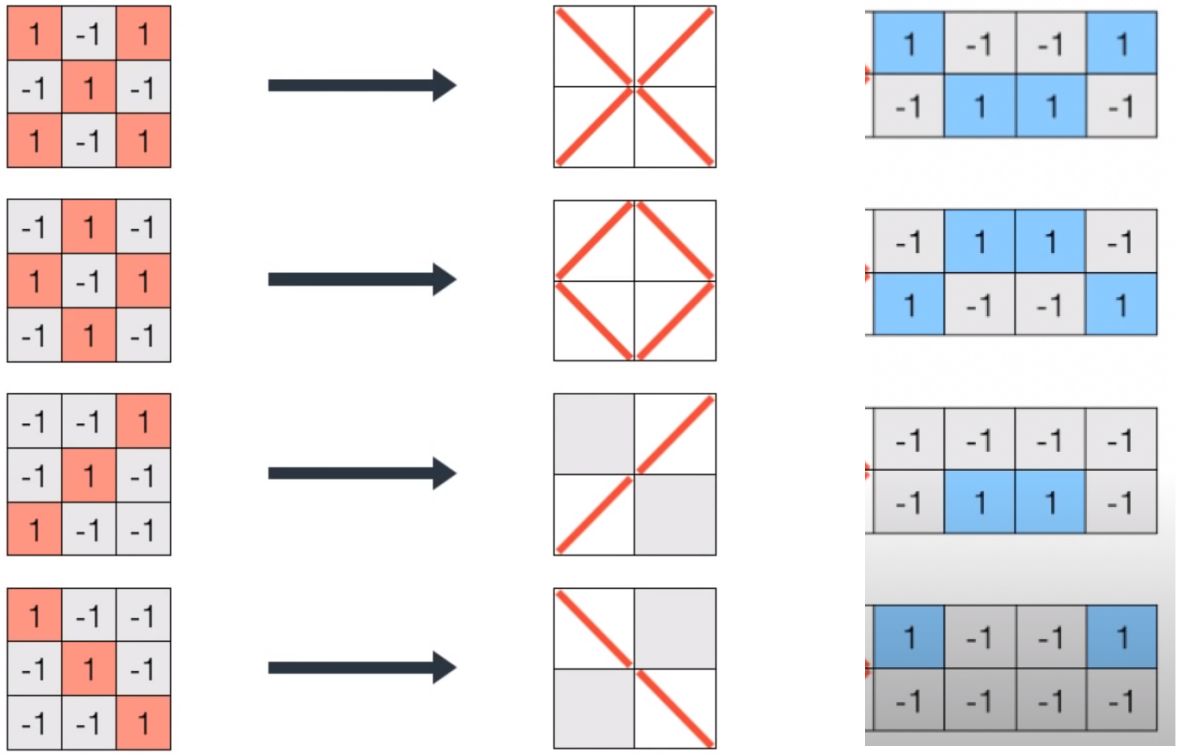



1 -1 1
-1 1 -1
1 -1 1
+ -
- +
4 -4
-4 4
- +
+ -
-4 4
4 -4
Convolution Layer
Pooling Layer


1 -1 -1 1


-1 1 1 -1

Recognition of characters: X , O, /, \



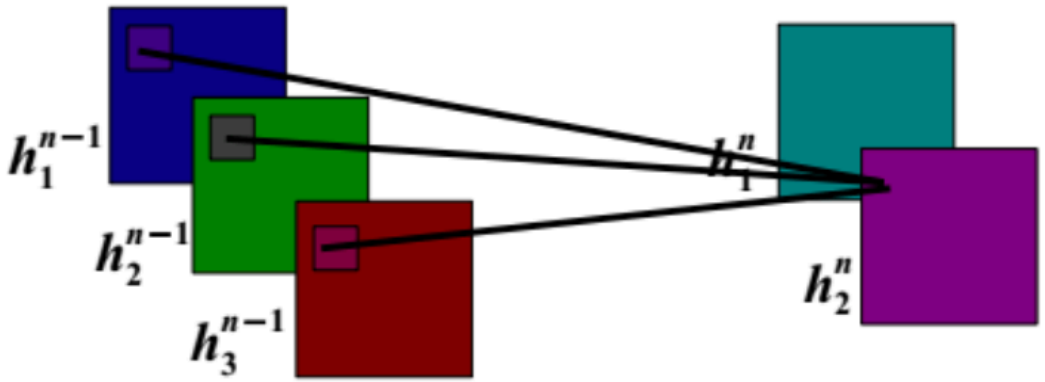
Convolutional Layer

$$h_j^n = max\left(0, \sum_{k=1}^{K} h_k^{n-1} * w_{kj}^n\right)$$

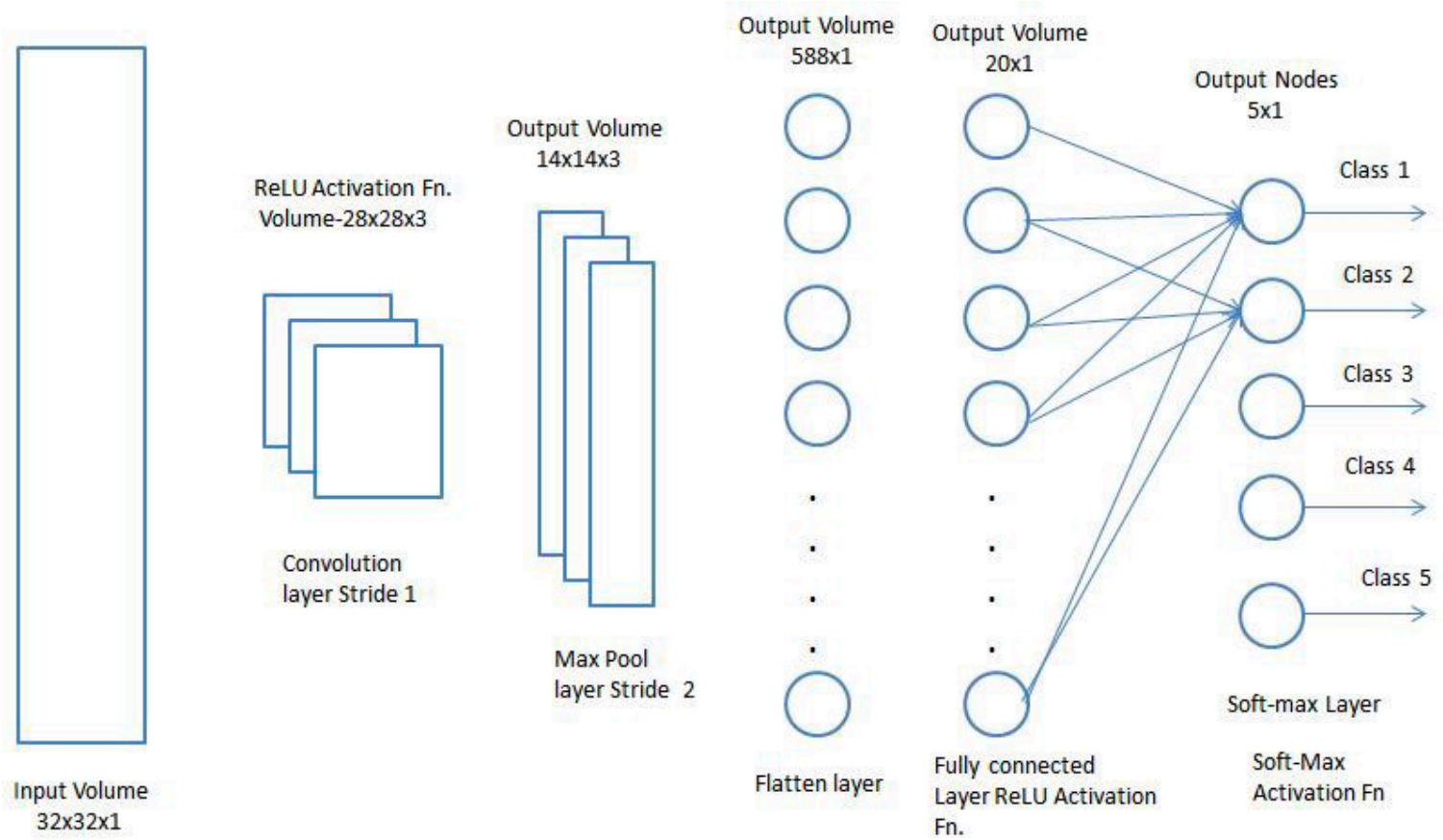
output feature map

input feature map

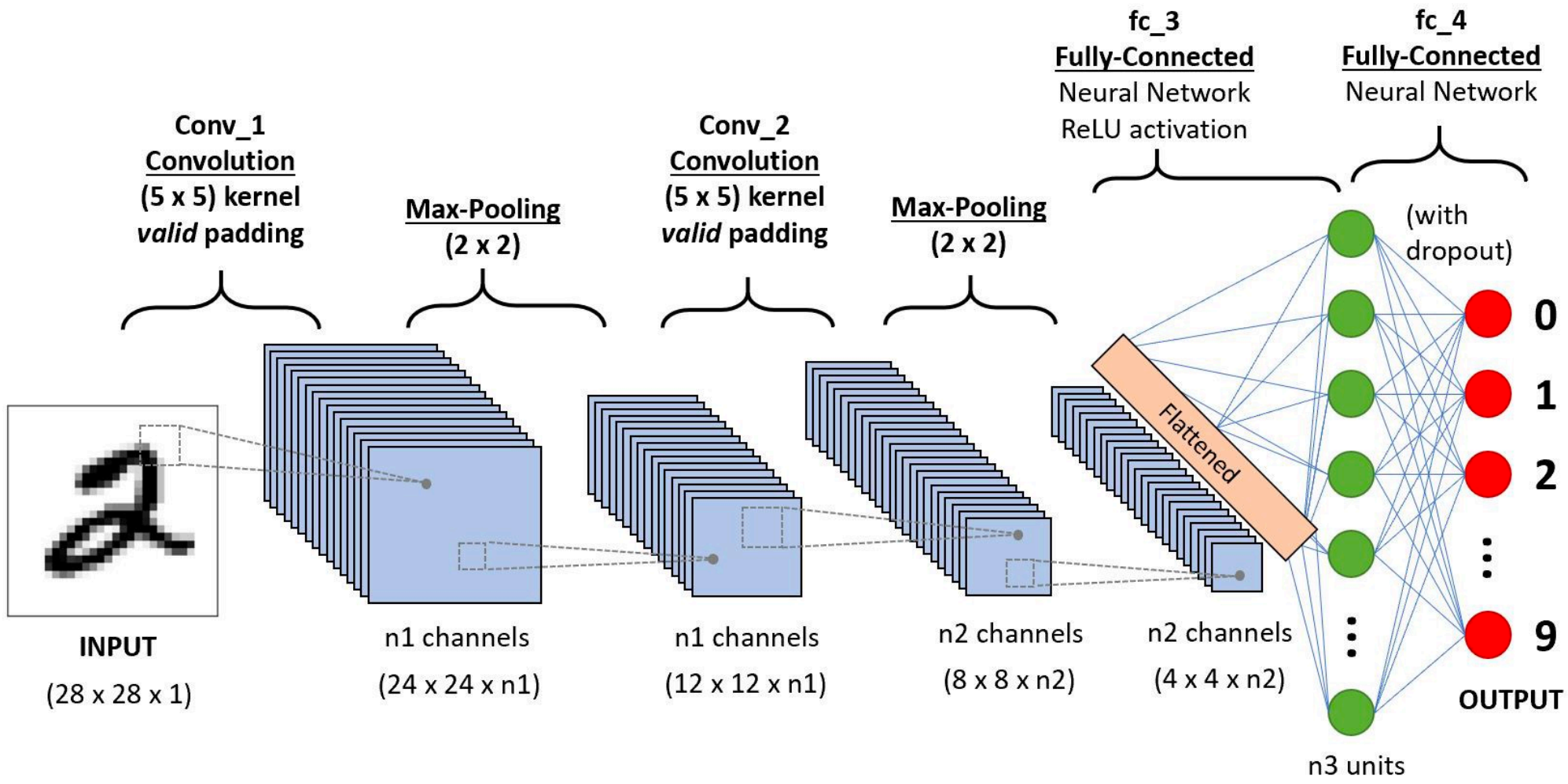
kernel

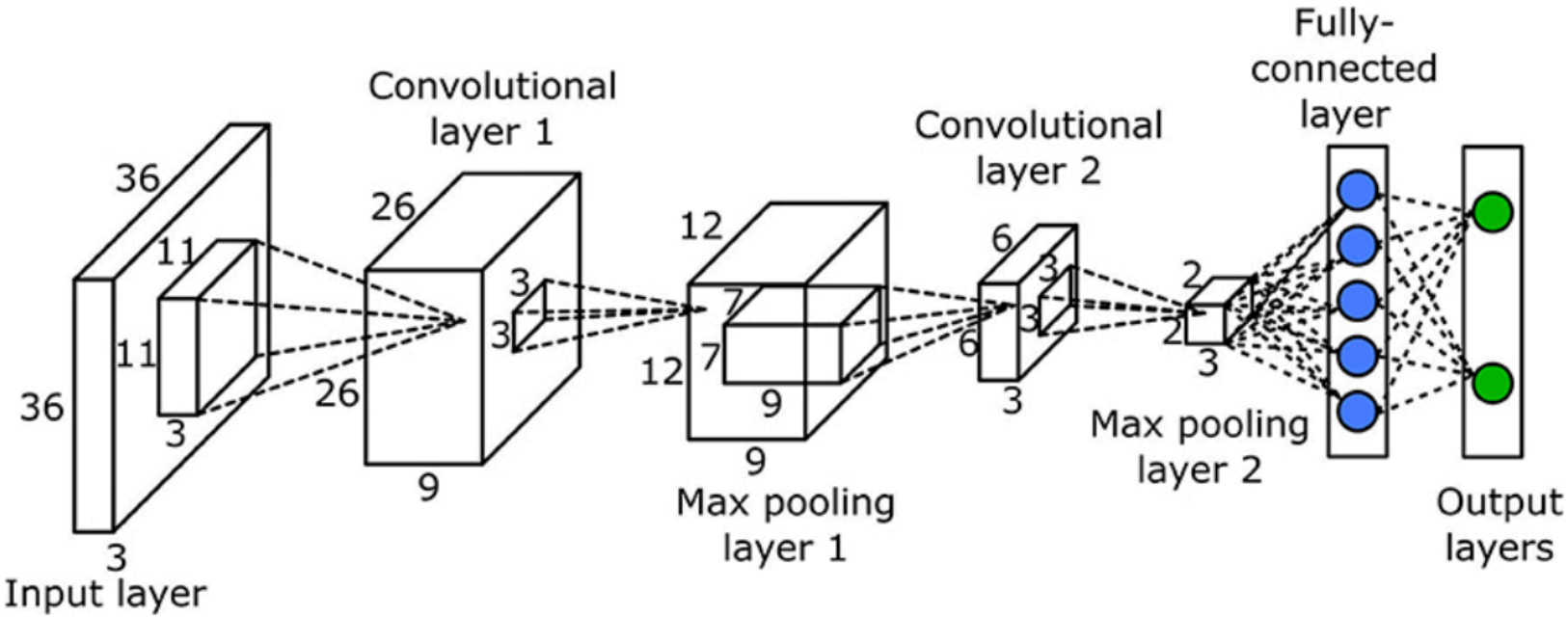


Classification — Fully Connected Layer (FC Layer)



Multi-Layer CNNs




36
11
11
36
3
3
Input layer
Convolutional
layer 1
26
3
3
26
9
12
7
7
12
9
9
Max pooling
layer 1
Convolutional
layer 2
6
3
3
6
3
Max pooling
layer 2
2
2
3
Fully-
connected
layer
Output
layers

Well known CNNs

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future.

1. LeNet
2. AlexNet
3. VGGNet
4. GoogLeNet
5. ResNet
6. ZFNet

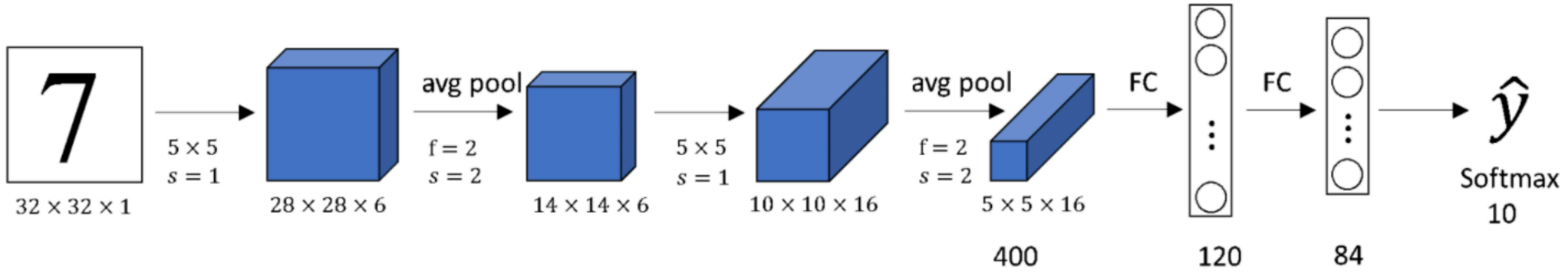
LeNet

LeNet is a convolutional neural network structure proposed by Yann LeCun et al. in 1998. In general, LeNet refers to lenet-5 and is a simple convolutional neural network.

LeNet5 was one of the earliest convolutional neural networks and promoted the development of deep learning. Since 1988, after years of research and many successful iterations, the pioneering work has been named LeNet5.

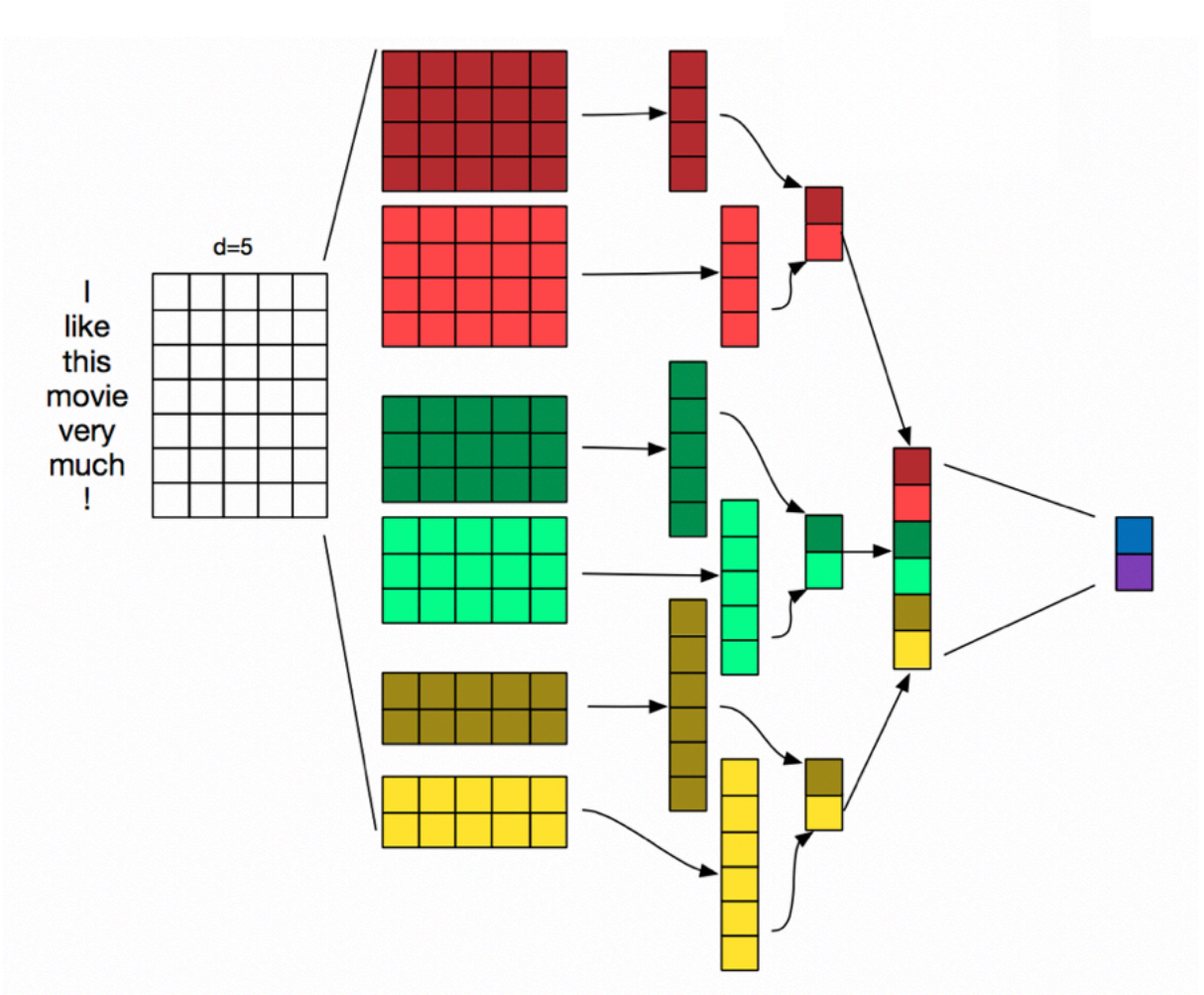
LeNet-5

http://datahacker.rs/deep-learning-lenet-5-architecture/



The goal of $LeNet{-}5$ was to recognize handwritten digits. So, it takes as an input $32{\times}32{\times}1$ image. It is a grayscale image, thus the number of channels is 1. Here is a picture of it's arhitecture. In the first step we use 65×5 filters with a stride s=1 and $nopadding$. Therefore we end up with a 28×28×6 volume. Notice that, because we are using s=1 and $nopadding$, the image dimensions reduce from 32×32 to 28×28.
Next, $LeNet$ applies $pooling$. When this paper was written $Averagepooling$ was much more in use, so here we will use $Averagepooling$. However, nowadays we would probably use $Maxpooling$ instead. So, here we will implement $Averagepool$ with filter f=2 and stride s=2. We get a 14×14×6 volume, so we reduced dimensions of an image by a factor of 2 and due to use of a stride of 2.

CNN for Natural Language Processing



CNN Implementation

```
model = Sequential()

model.add(Conv2D(32, (3, 3), input_shape = (64, 64, 3), activation = 'relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size = (2, 2)))

model.add(Flatten())

model.add(Dense(units = 128, activation = 'relu'))
model.add(Dense(units = 1, activation = 'sigmoid'))

model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics = ['accuracy'])
```

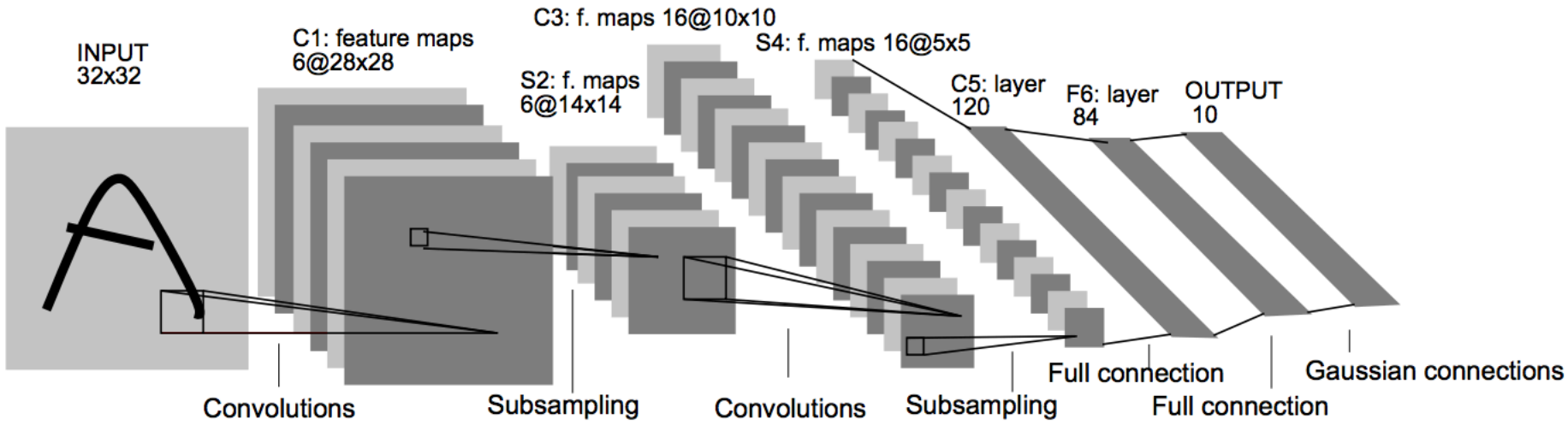
Topics

- Classic network architectures (for historical purposes)
- LeNet-5
- AlexNet
- VGG 16
- Modern network architectures
- Inception
- ResNet
- ResNeXt
- DenseNet

LeNet-5

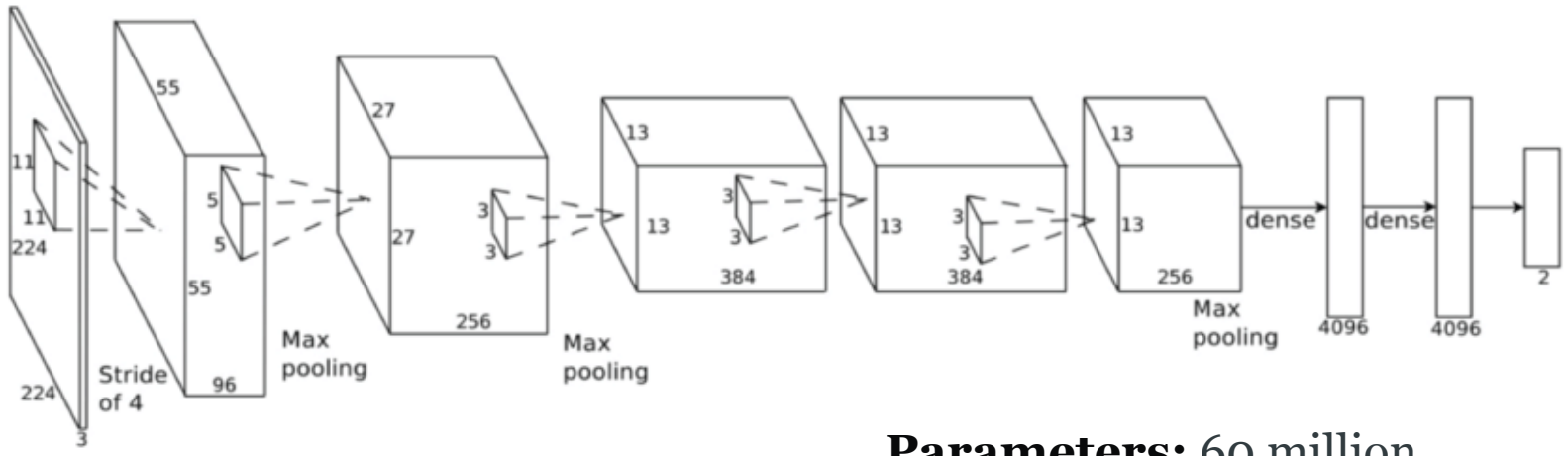
Yann Lecun's LeNet-5 model was developed in 1998 to identify handwritten digits for zip code recognition in the postal service. This pioneering model largely introduced the convolutional neural network as we know it today.

Parameters: 60,000



AlexNet

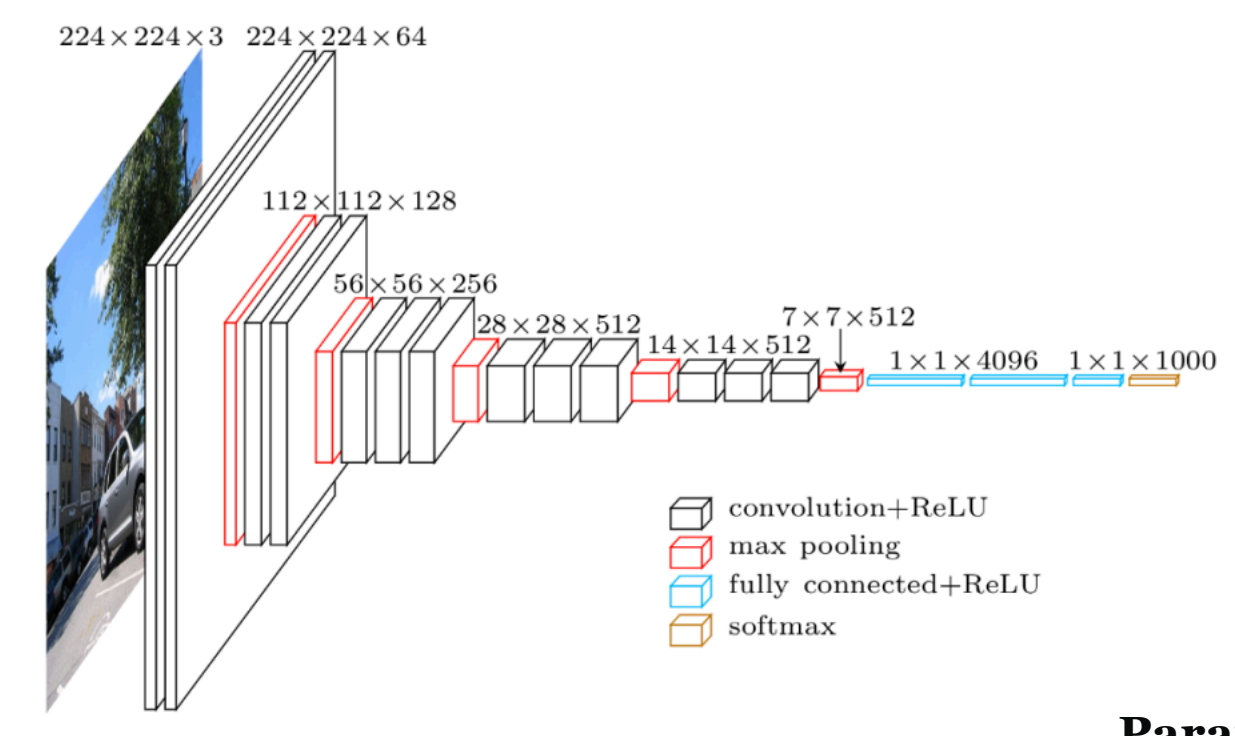
AlexNet was developed by Alex Krizhevsky et al. in 2012 to compete in the ImageNet competition. The general architecture is quite similar to LeNet-5, although this model is considerably larger. The success of this model (which took first place in the 2012 ImageNet competition) convinced a lot of the computer vision community to take a serious look at deep learning for computer vision tasks.



Parameters: 60 million

VGG16

The VGG network, introduced in 2014, offers a deeper yet simpler variant of the convolutional structures discussed above. At the time of its introduction, this model was considered to be very deep.

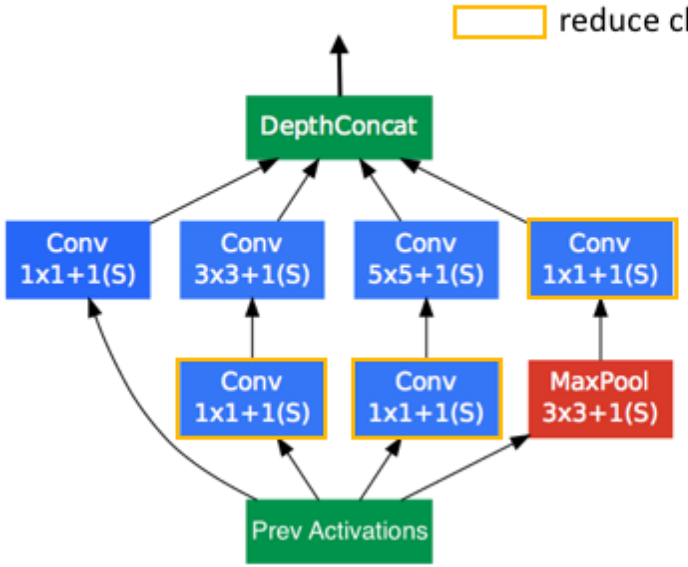


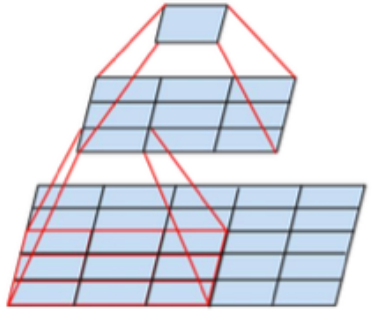
Parameters: 138 million

Inception (GoogLeNet)

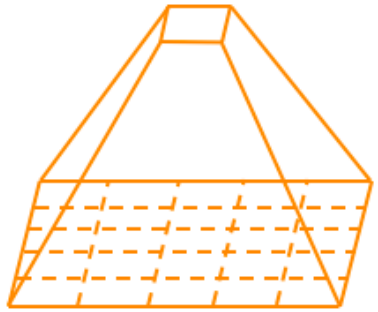
In 2014, researchers at Google introduced the Inception network which took first place in the 2014 ImageNet competition for classification and detection challenges.
The model is comprised of a basic unit referred to as an "Inception cell" in which we perform a series of convolutions at different scales and subsequently aggregate the results. In order to save computation, 1x1 convolutions are used to reduce the input channel depth. For each cell, we learn a set of 1x1, 3x3, and 5x5 filters which can learn to extract features at different scales from the input. Max pooling is also used, albeit with "same" padding to preserve the dimensions so that the output can be properly concatenated.

Inception (GoogLeNet)

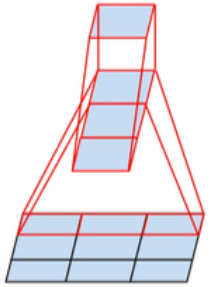




two successive 3x3 convolutions



5x5 convolution

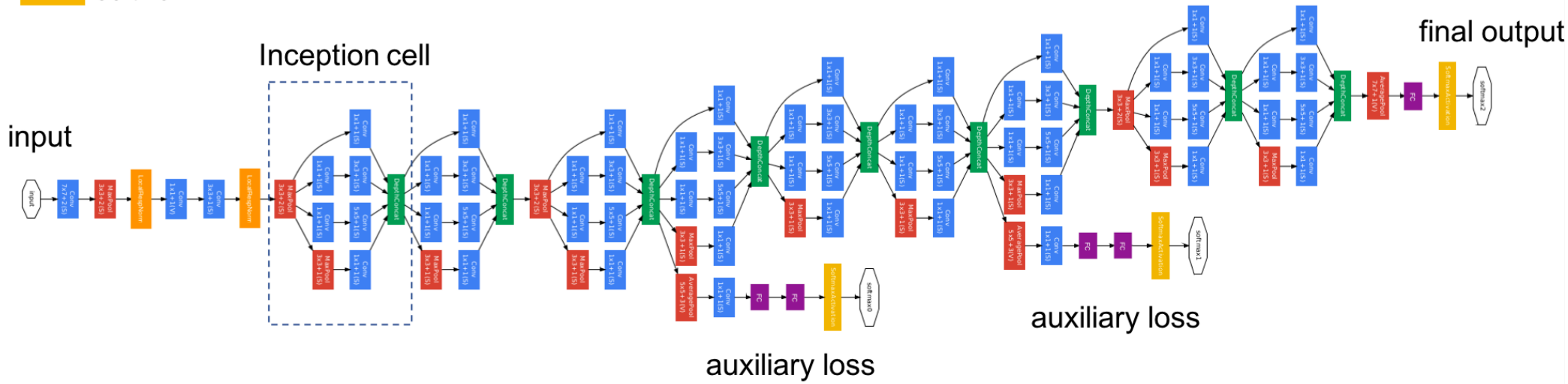


Whereas a 5×5×c5×5×□ filter requires 25c25□ parameters, two 3×3×c3×3×□ filters only require 18c18□ parameters. In order to most accurately represent a 5x5 filter, we shouldn't use any nonlinear activations between the two 3x3 layers. However, it was discovered that "linear activation was always inferior to using rectified linear units in all stages of the factorization."
It was also shown that 3x3 convolutions could be further deconstructed into successive 3x1 and 1x3 convolutions.

Inception (GoogLeNet)

convolution
max pooling
channel concatenation
channel-wise normalization
fully-connected layer
softmax

Parameters: 5 million (V1) and 23 million (V3)

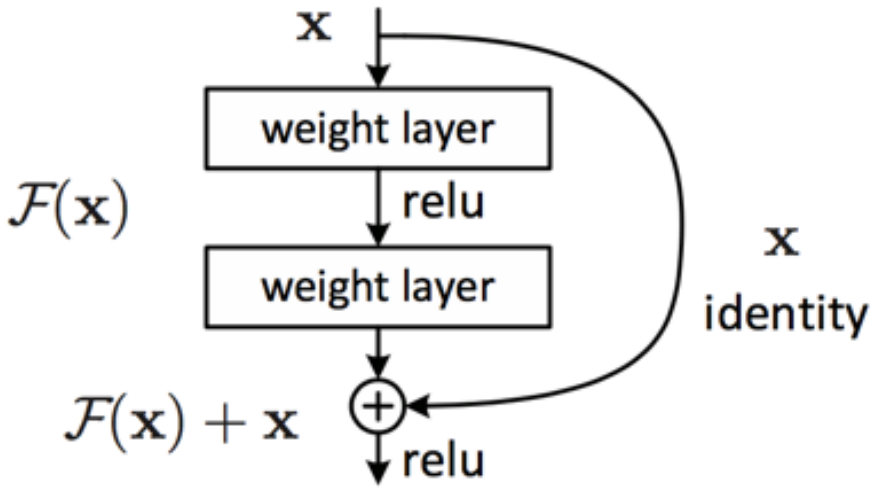


ResNet

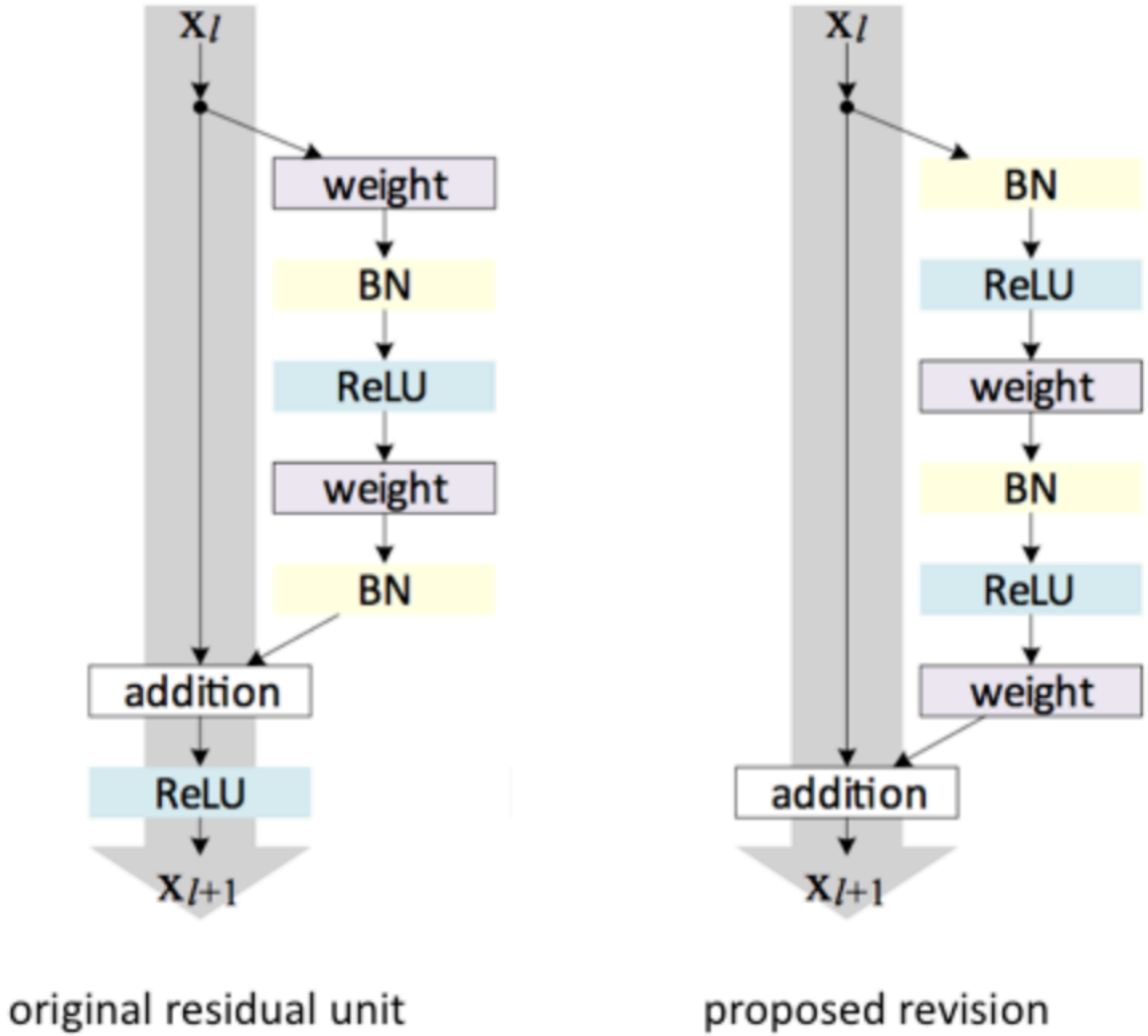
- Deep residual networks were a breakthrough idea which enabled the development of much deeper networks (hundreds of layers as opposed to tens of layers).
- Its a generally accepted principle that deeper networks are capable of learning more complex functions and representations of the input which should lead to better performance. However, many researchers observed that adding more layers eventually had a negative effect on the final performance.

> *Let us consider a shallower architecture and its deeper counterpart that adds more layers onto it. There exists a solution* ***by construction*** *to the deeper model: the added layers are* ***identity*** *mapping, and the other layers are copied from the learned shallower model. The existence of this constructed solution indicates that* ***a deeper model should produce no higher training error than its shallower counterpart****. But experiments show that our current solvers on hand are unable to find solutions that are comparably good or better than the constructed solution (or unable to do so in feasible time).*

ResNet


x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

ResNet



original residual unit

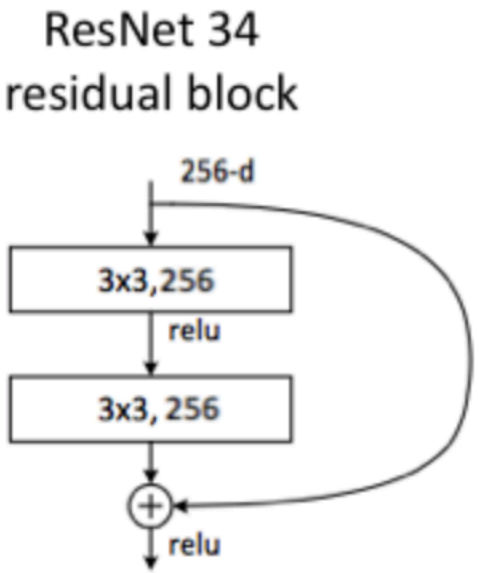
proposed revision

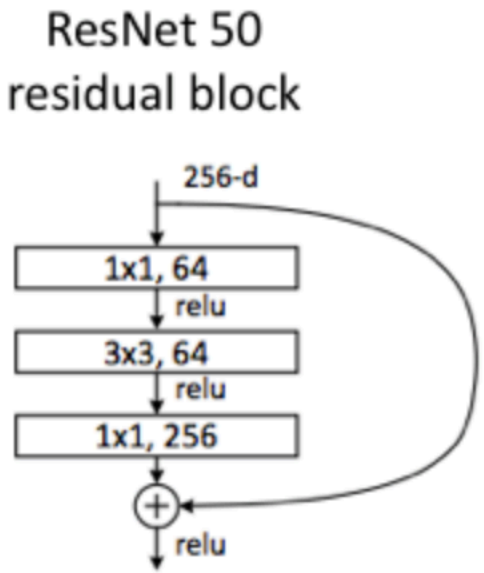
ResNet

Wide residual networks

Although the original ResNet paper focused on creating a network architecture to enable deeper structures by alleviating the degradation problem, other researchers have since pointed out that increasing the network's width (channel depth) can be a more efficient way of expanding the overall capacity of the network.

RestNet





Parameters: 25 million (ResNet 50)

ResNeXt

The ResNeXt architecture is an extension of the deep residual network which replaces the standard residual block with one that leverages a "*split-transform-merge*" strategy (ie. branched paths within a cell) used in the Inception models. Simply, rather than performing convolutions over the full input feature map, the block's input is projected into a series of lower (channel) dimensional representations of which we separately apply a few convolutional filters before merging the results.

DenseNet

The idea behind dense convolutional networks is simple: **it may be useful to reference feature maps from earlier in the network**. Thus, each layer's feature map is concatenated to the input of *every successive layer* within a dense block. This allows later layers within the network to *directly* leverage the features from earlier layers, encouraging feature reuse within the network. The authors state, "concatenating feature-maps learned by *different layers* increases variation in the input of subsequent layers and improves efficiency."

THANK YOU!