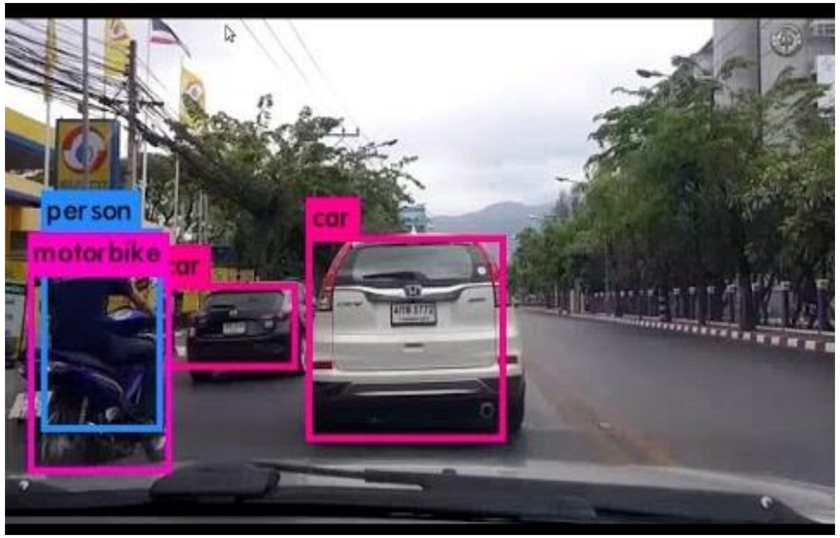
Convolutional Neural Network

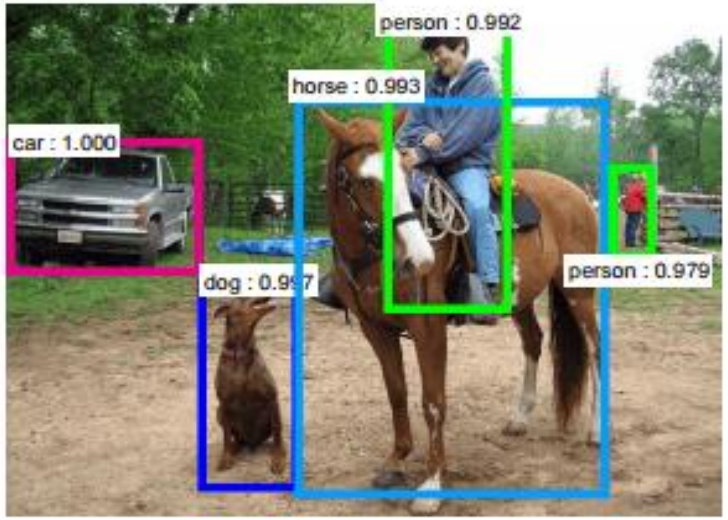
Face Recognition

Style Transferring

Image quality enhancement
Beautification



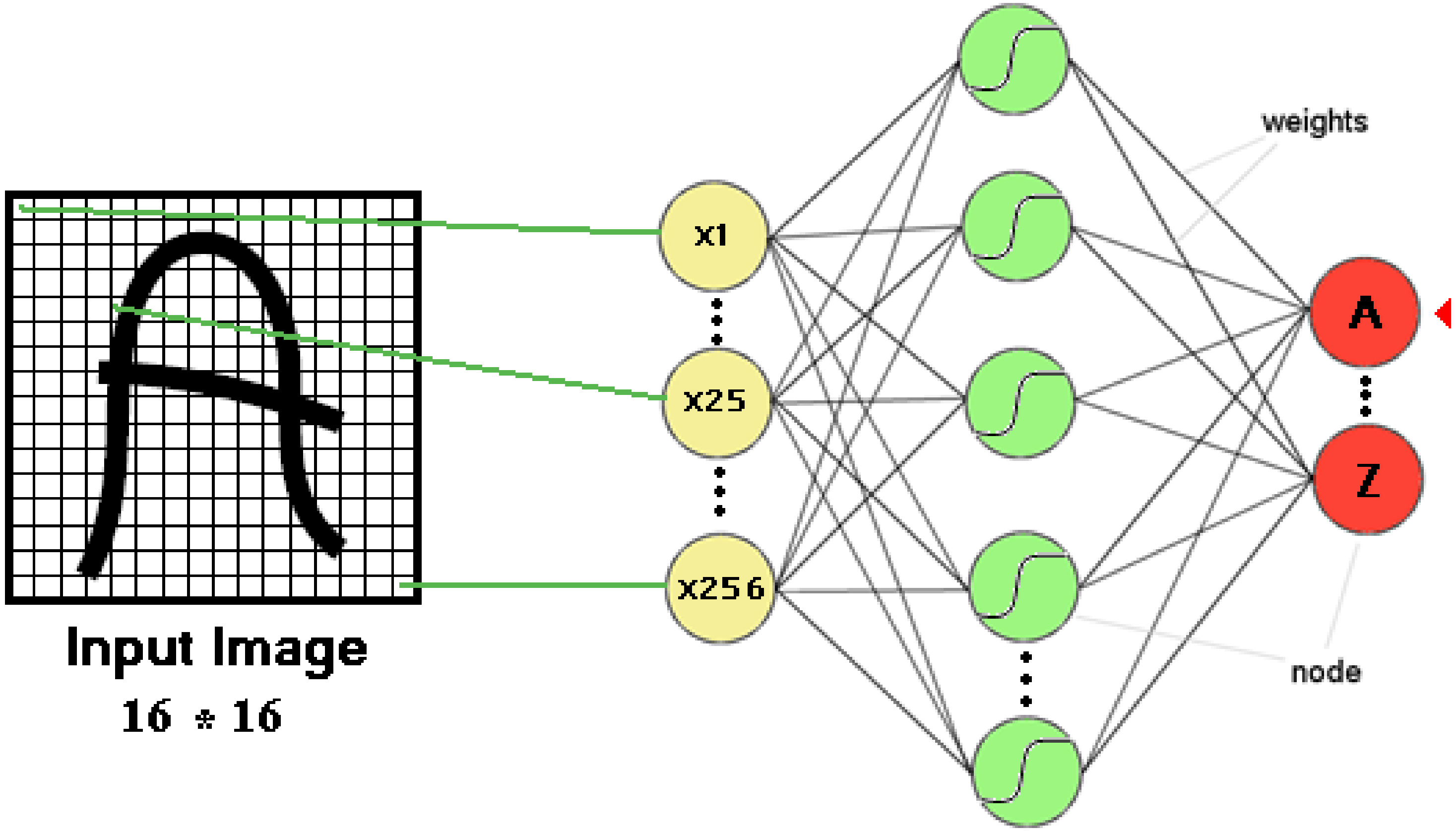
Object detection (Self driving car)



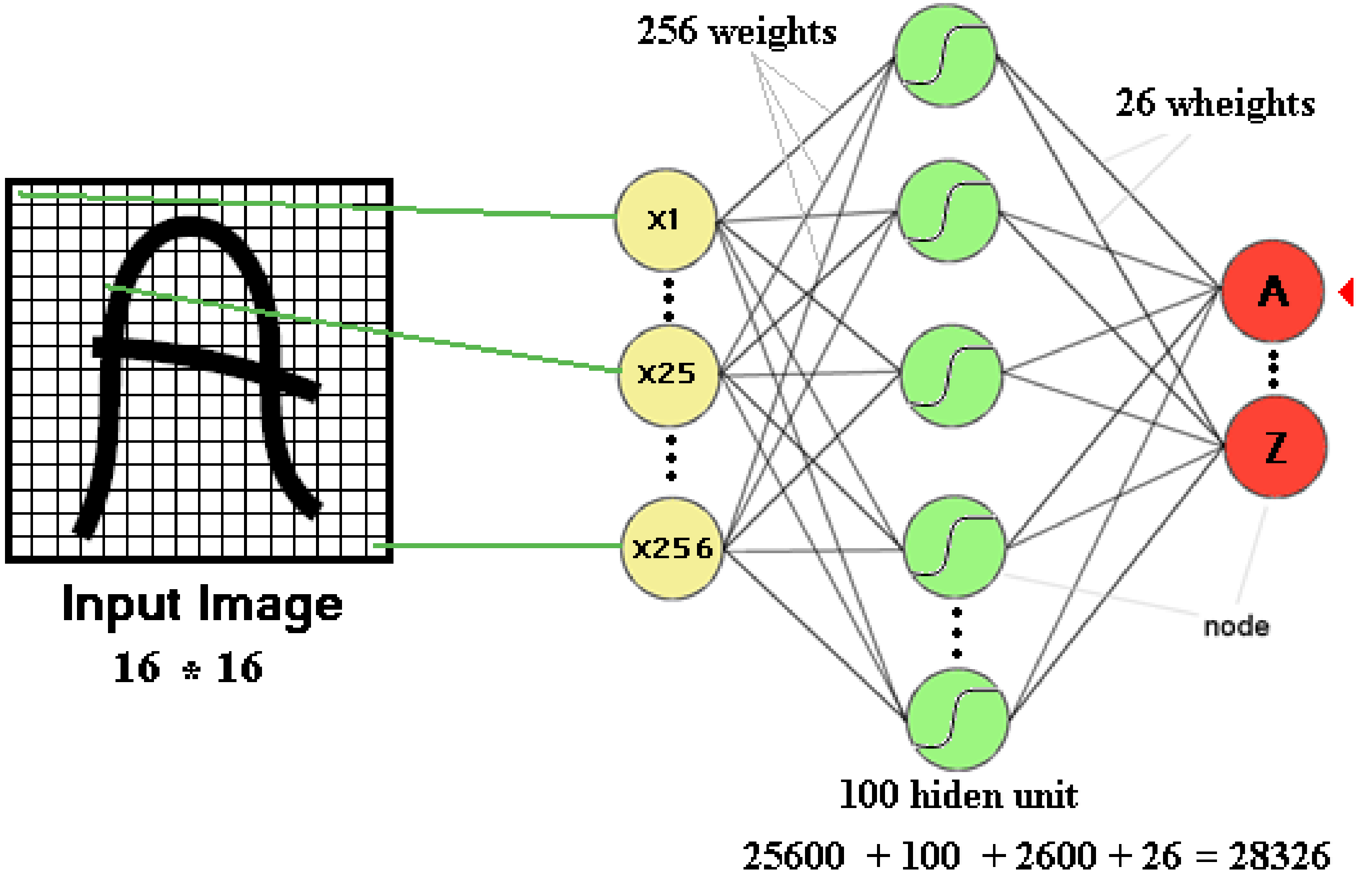
Classification

Gesture Recognition

Multi-layer perceptron and image processing



16x16 image

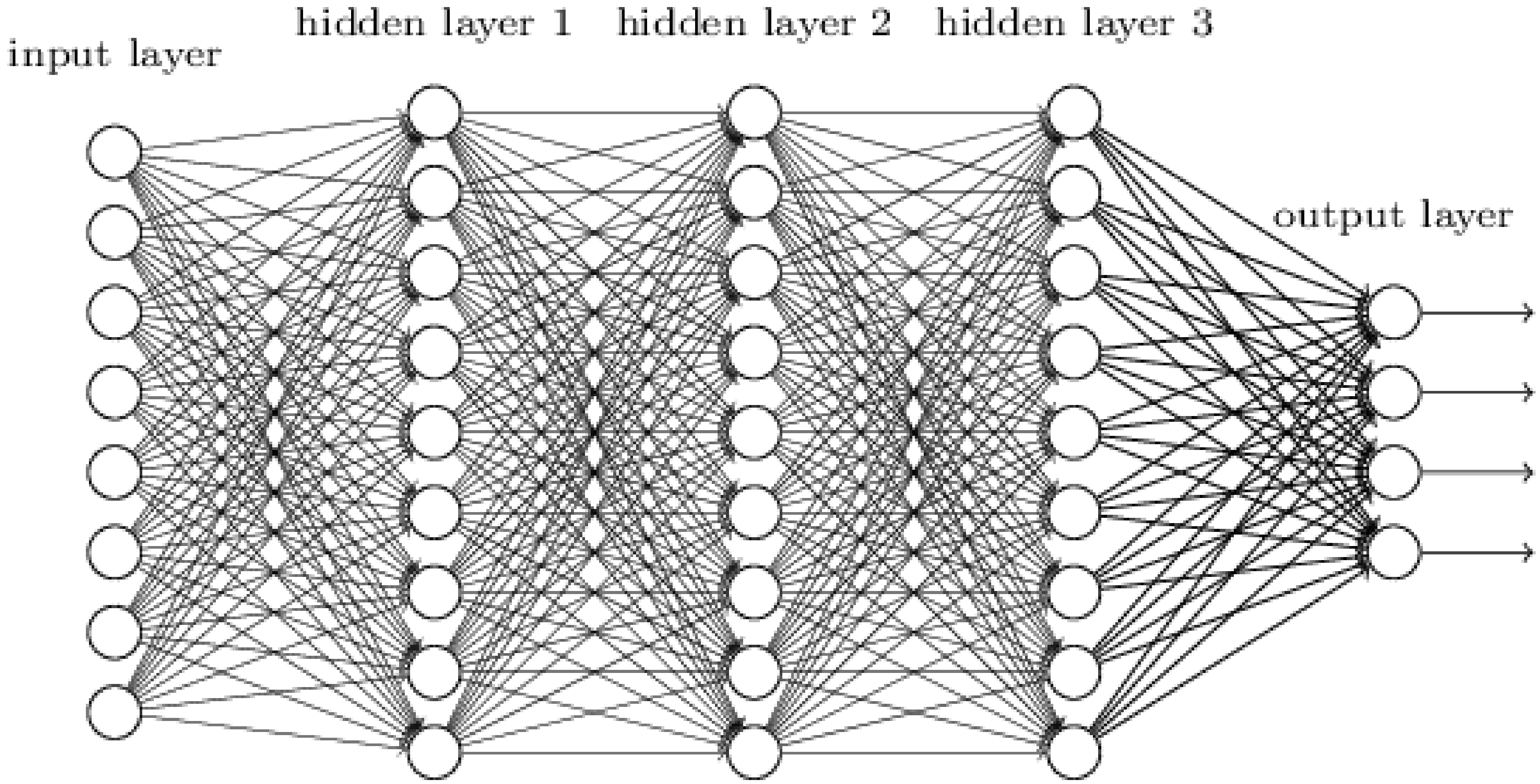


256x100 + 100 bias + 100x26 output neurons + 26 bias = 28236

The number of trainable parameters becomes extremely large

ANN: Too many parameters

- We know it is good to learn a small model.
- From this fully connected model, do really need all the edges?
- Can some of these be shared?



Identify

Can we do with less information?

How much information we can throw away and still recognize the object?

10%

20%

Can we do with less information?

How much information we can throw away and still recognize the object?

10%
20%
50%
75%

Subsampling pixels will not change the object

bird

Subsampling

bird

We can subsample the pixels to make image smaller fewer parameters to characterize the image


TIPO EXCELSIUS

CNNs Vs. ANNs

- ANNs suffer from curse of dimensionality when it comes to high resolution images

- We use filters (receptive fields) to exploit spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers

  - *Parameter Sharing*
  - *Sparsity of connection*

Convolution

- Convolution is a pointwise multiplication of two functions to produce a third function.

- Primary purpose of convolution in CNN is to extract features from the input image.

- Matrix formed by sliding the filter over the image and computing the dot product is called the ‘Convolved Feature’ or ‘Activation Map’ or the ‘Feature Map’.

Convolution Example

3_{1}	0_{0}	1_{-1}	2	7	4
1_{1}	5_{0}	8_{-1}	9	3	1
2_{1}	7_{0}	2_{-1}	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6X6 Matrix (nXn)

*

Convolution

1	0	-1
1	0	-1
1	0	-1

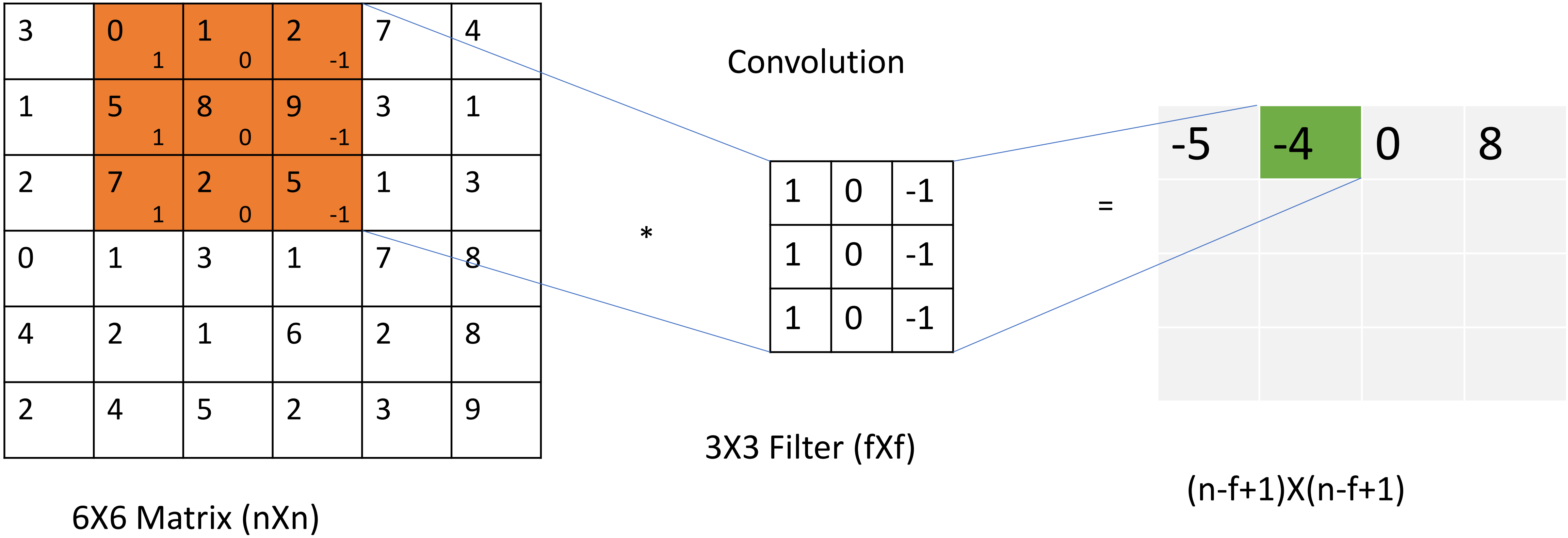
3X3 Filter (fXf)

=

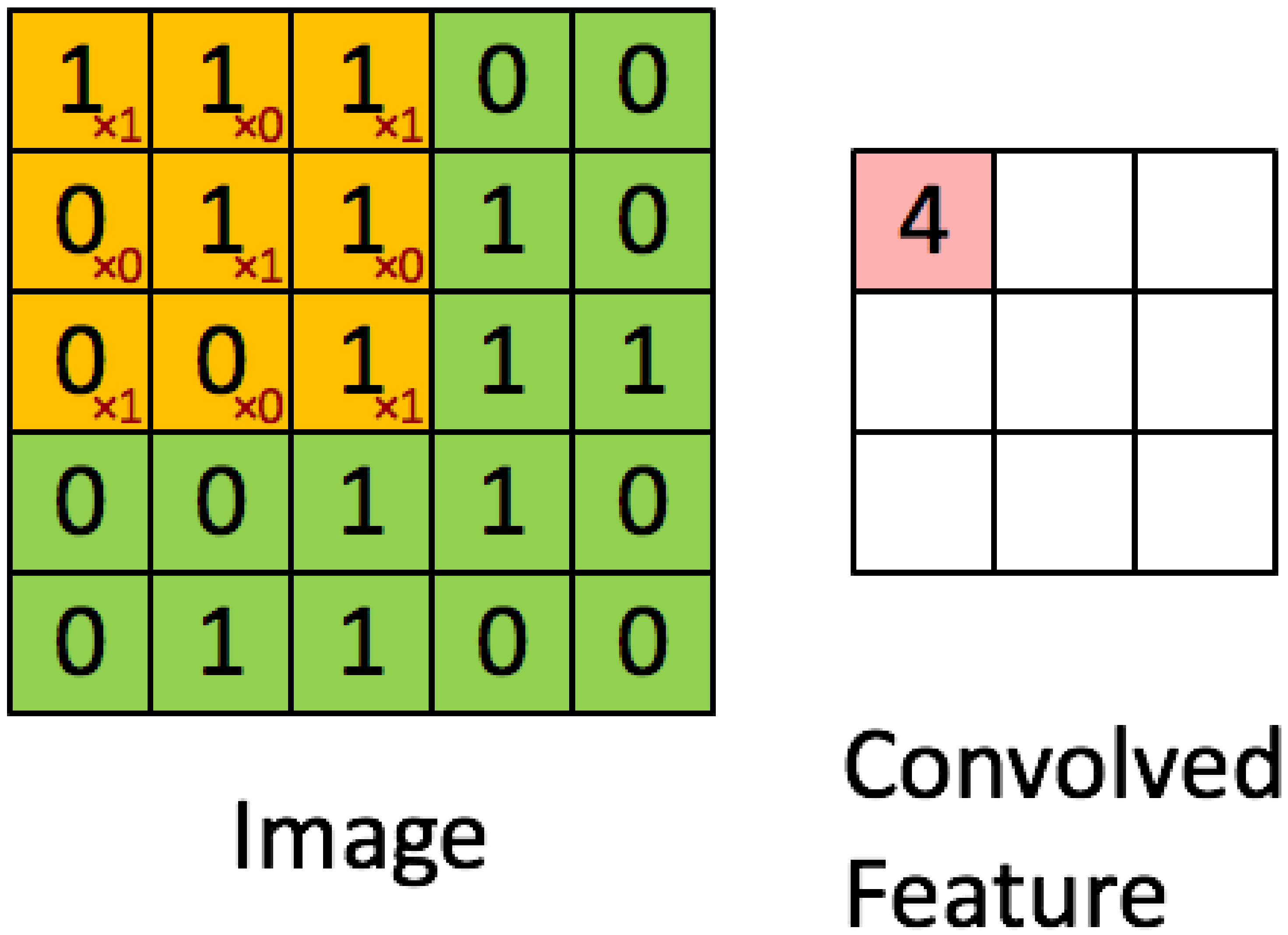
-5	-4	0	8

(n-f+1)X(n-f+1)

Convolution Example



Convolution Example



*http://ufldl.stanford.edu/tutorial/supervised/FeatureExtractionUsingConvolution/

Detecting Vertical edges

6*6 = 36

4*4=16

$$\begin{bmatrix} 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \\ 10 & 10 & 10 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}$$

- In case of ANN # parameter to train = 36*16 = 576
- In case of CNN # parameter to train = 9

Filter Weights

1	1	1
0	0	0
-1	-1	-1

Horizontal Filter

1	0	-1
2	0	-2
1	0	-1

Sobel Filter

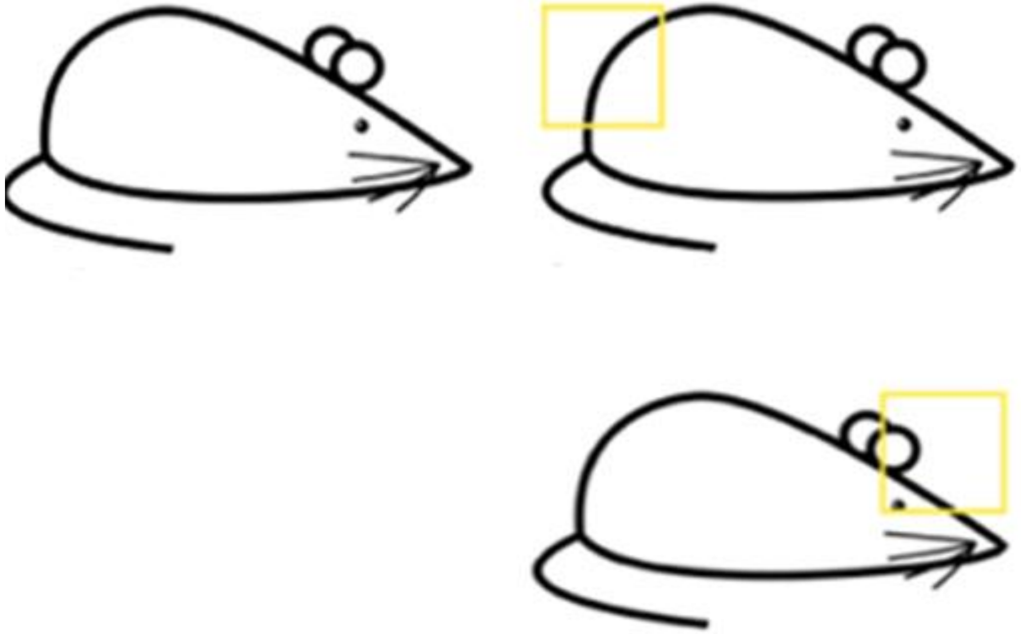
3	0	-3
10	0	-10
3	0	-3

Schorr Filter

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

Convolutional Neural Networks automatically estimates the weights of the filter

More Intuition



0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

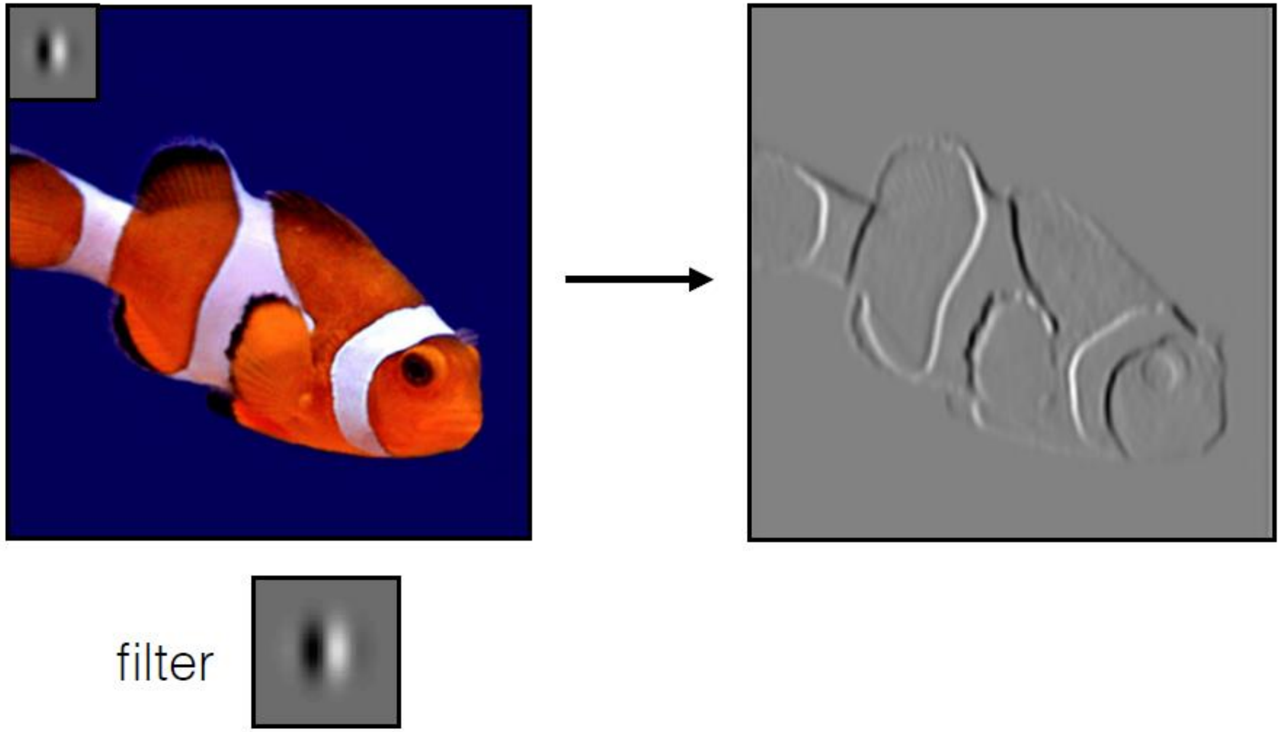
Pixel representation of a filter

Edge

Interpretation

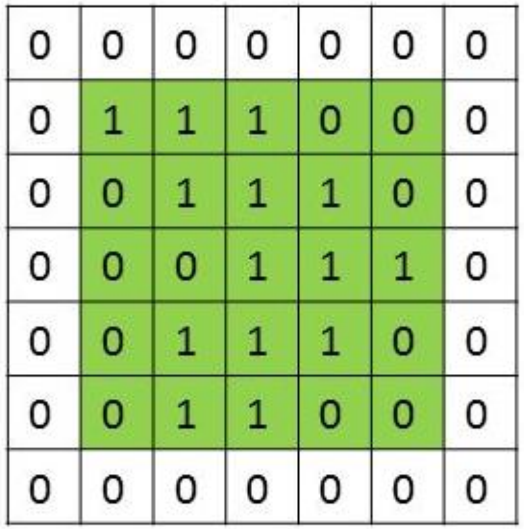
Convolution is just another way of computing W^TX

In CNN, **input** is **image**, **kernel** is **convolution filter** to be learned, **response** is the **feature map**

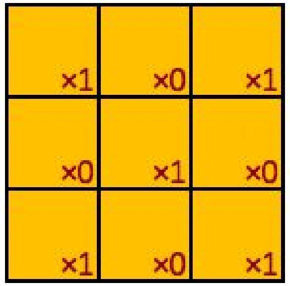


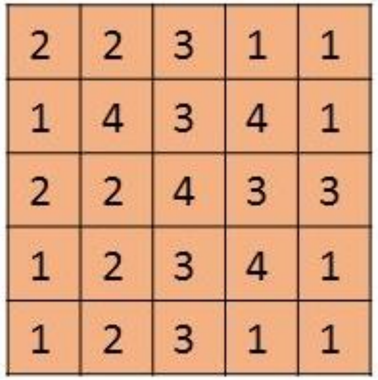
Padding

- Padding is used to preserve the original dimensions of the input
- Zeros are added to outside of the input
- Number of zero layers depend upon the size of the kernel



5X5 (with padding)

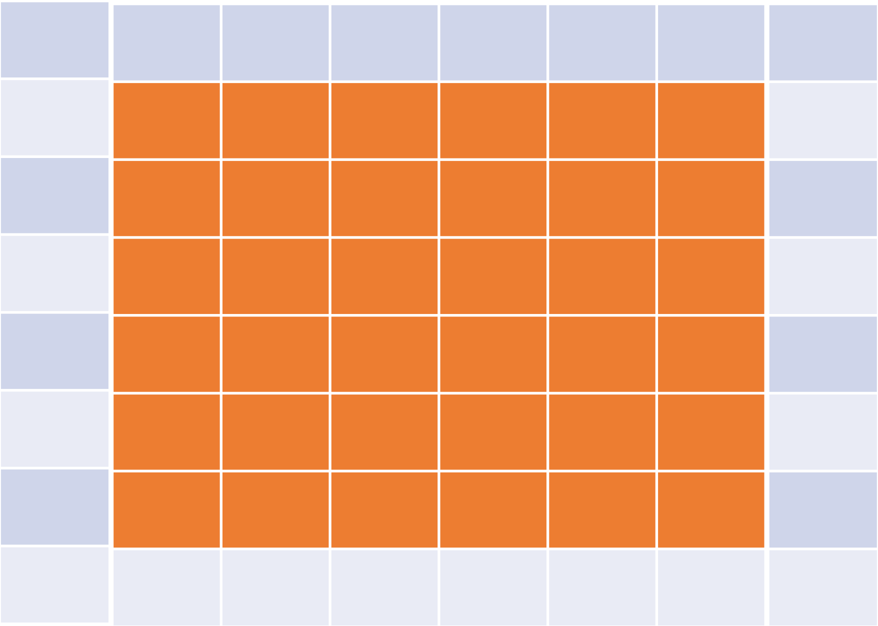




5X5

Padding

Stride=s

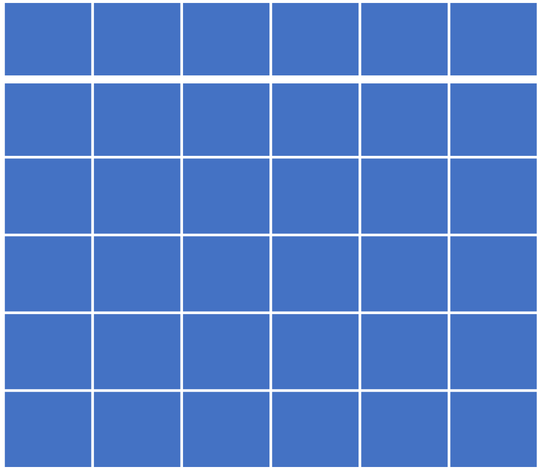


nXn 6X6 to 8X8 Padding=1

*

f 3X3

=



(n-f+1)X(n-f+1) to (n+2p-f+1)X(n+2p-f+1)

Valid to same

$$\text{Floor}\left(\frac{n+2p-f}{s}+1\right) X \ \text{Floor}\left(\frac{n+2p-f}{s}+1\right)$$

Stride

3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6X6 Matrix

1	0	-1
1	0	-1
1	0	-1

3X3 Filter

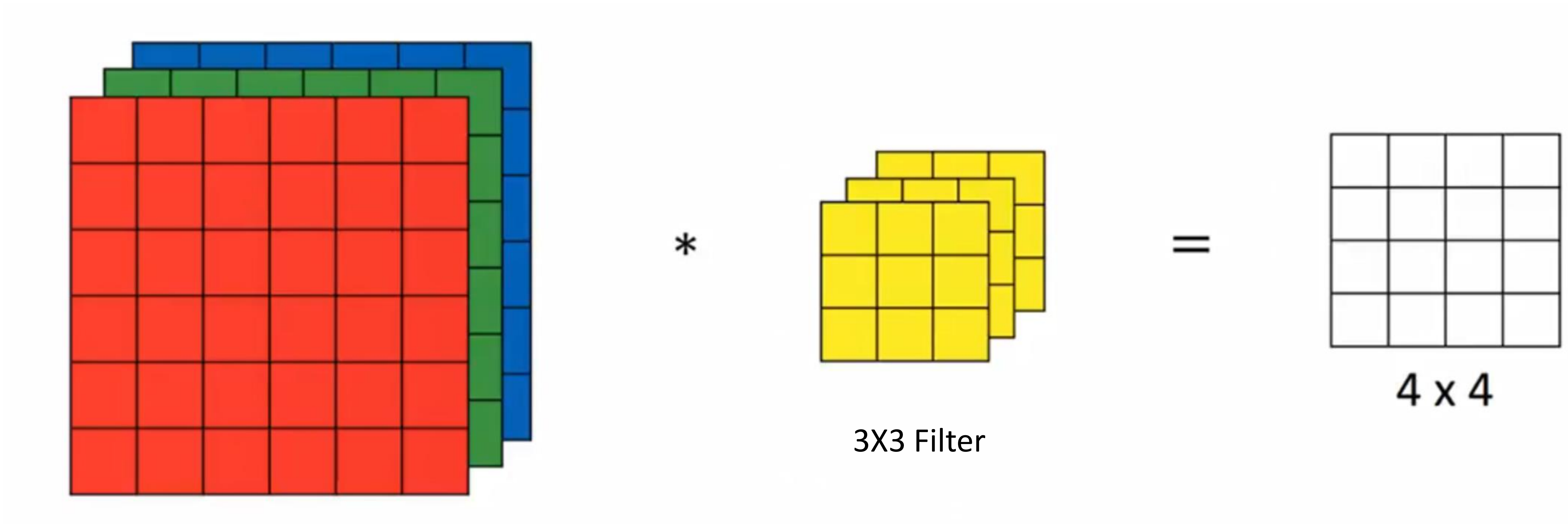
-5	8
-3	-16

2X2

Stride=s (3 Here)

Floor($\frac{n+2p-f}{s}+1$) XFloor($\frac{n+2p-f}{s}+1$)

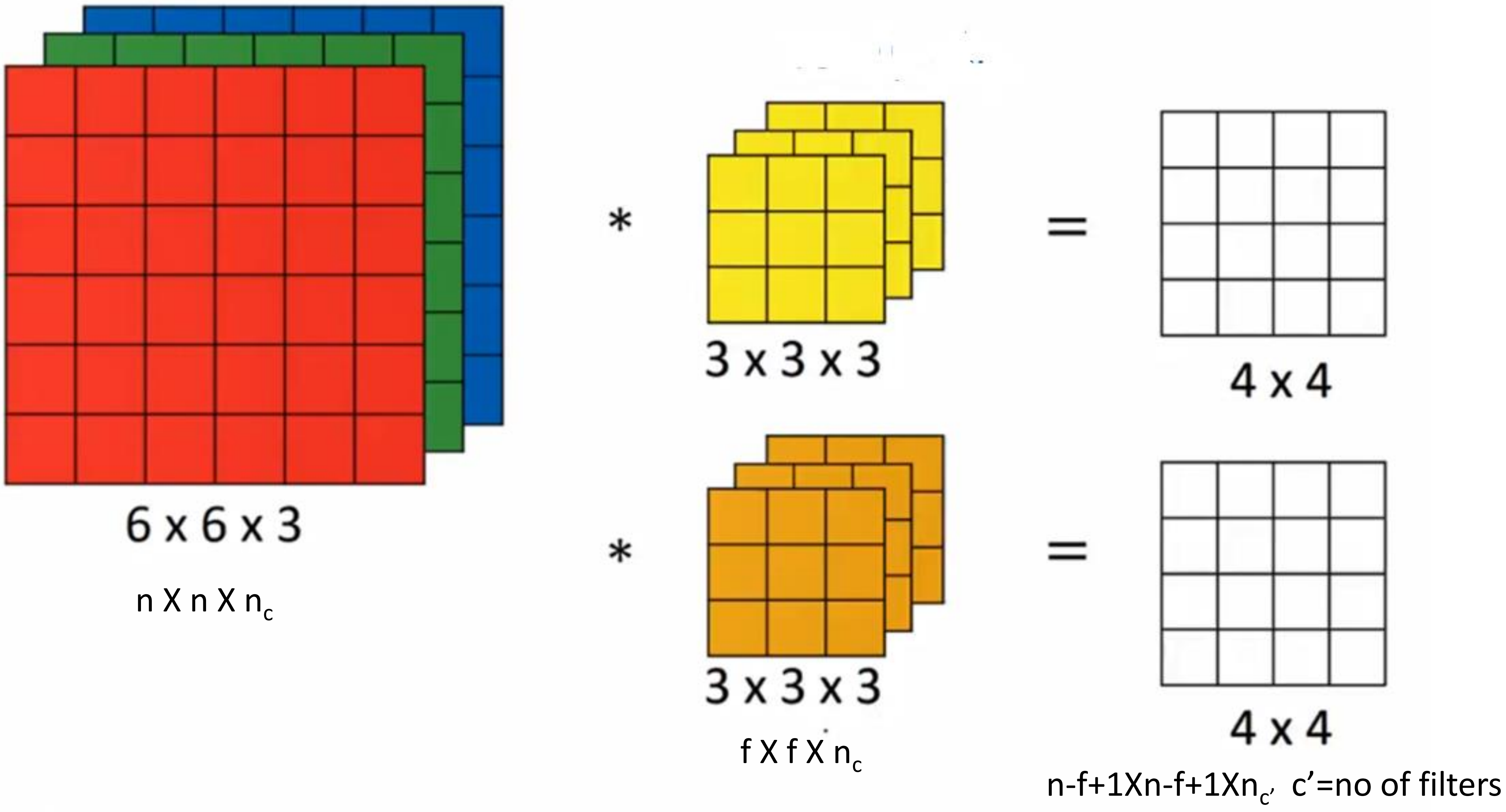
Channels



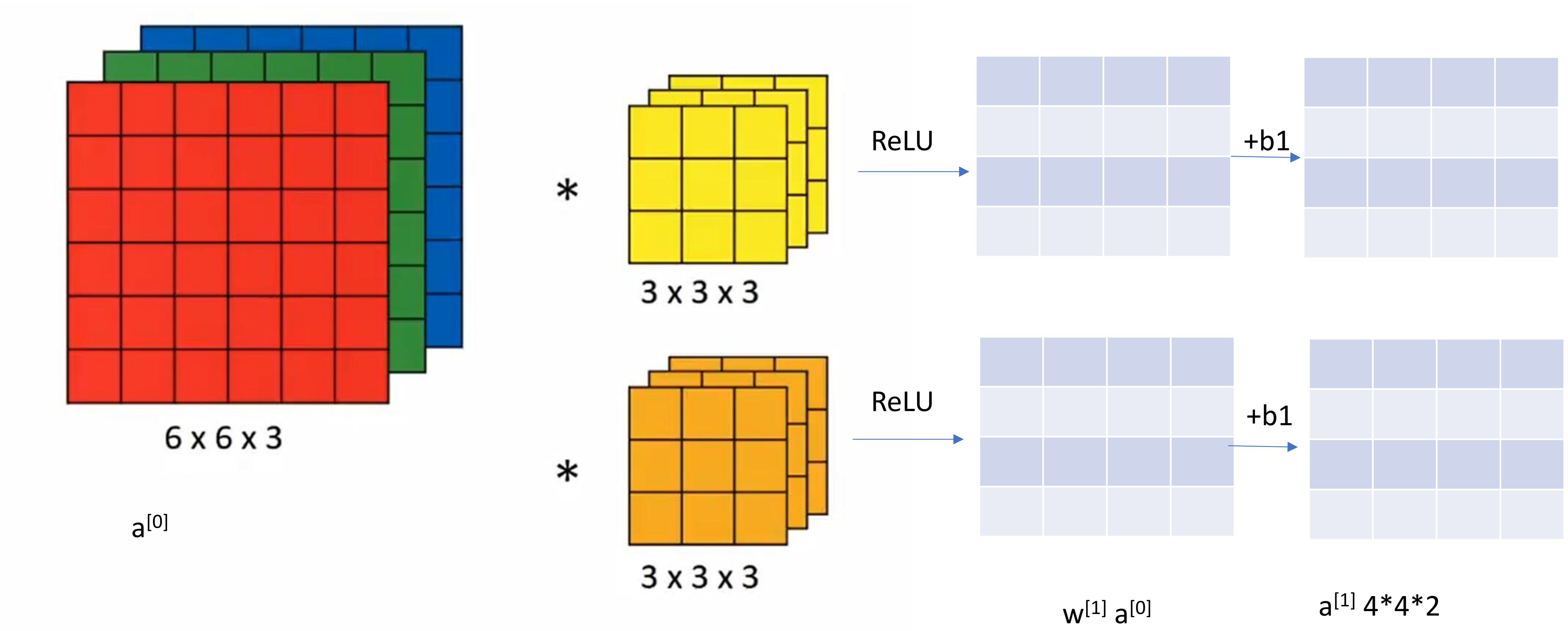
3X3 Filter

6X6 Matrix

Padding =0 Stride=1



*
*
=
=
6 x 6 x 3
3 x 3 x 3
3 x 3 x 3
4 x 4
4 x 4
n X n X n_c
f X f X n_c
n-f+1Xn-f+1X$n_{c'}$ c'=no of filters



$w^{[1]}$ 2 filters means two units here

Pooling

1	4	6	3
1	8	9	7
2	9	1	2
3	4	4	3

Max Pooling : One example of pooling layer

8	9
9	4

f=2
s=2   4X4 converted to 2X2

Function of Pooling is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network.

Pooling

1	4	6	3
1	8	9	7
2	9	1	2
3	4	4	3

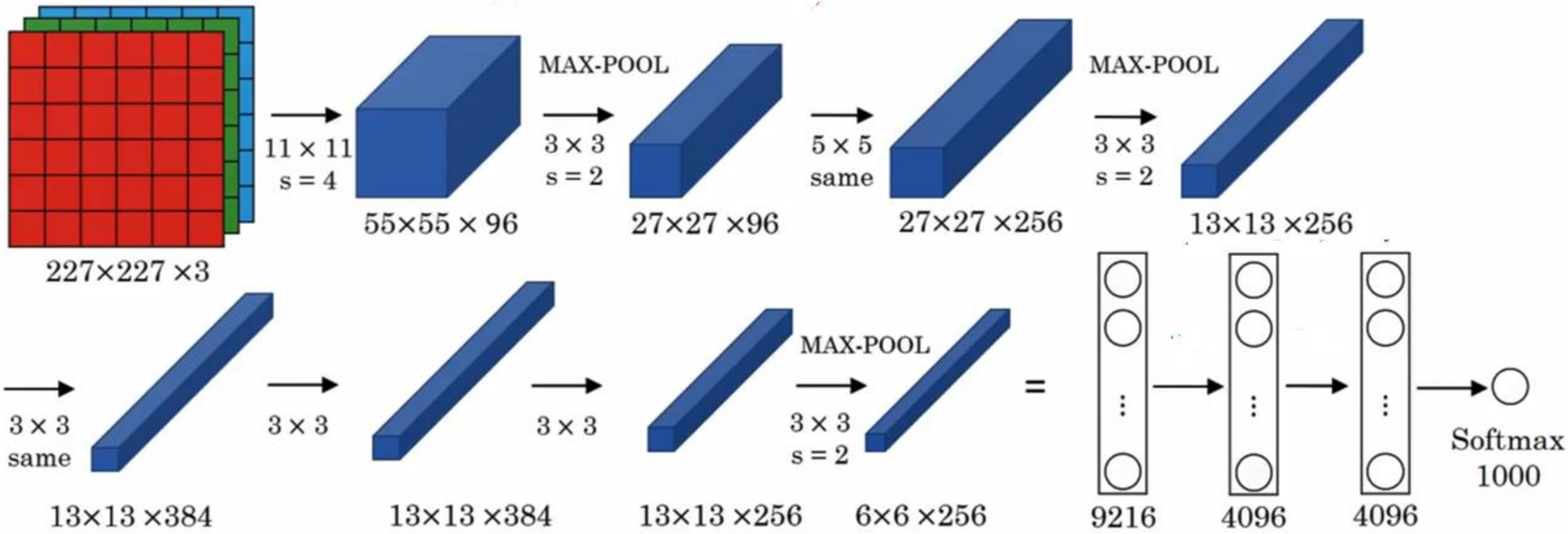
3.5	6.25
4.5	2.5

Average Pooling : Another example of pooling layer

f=2

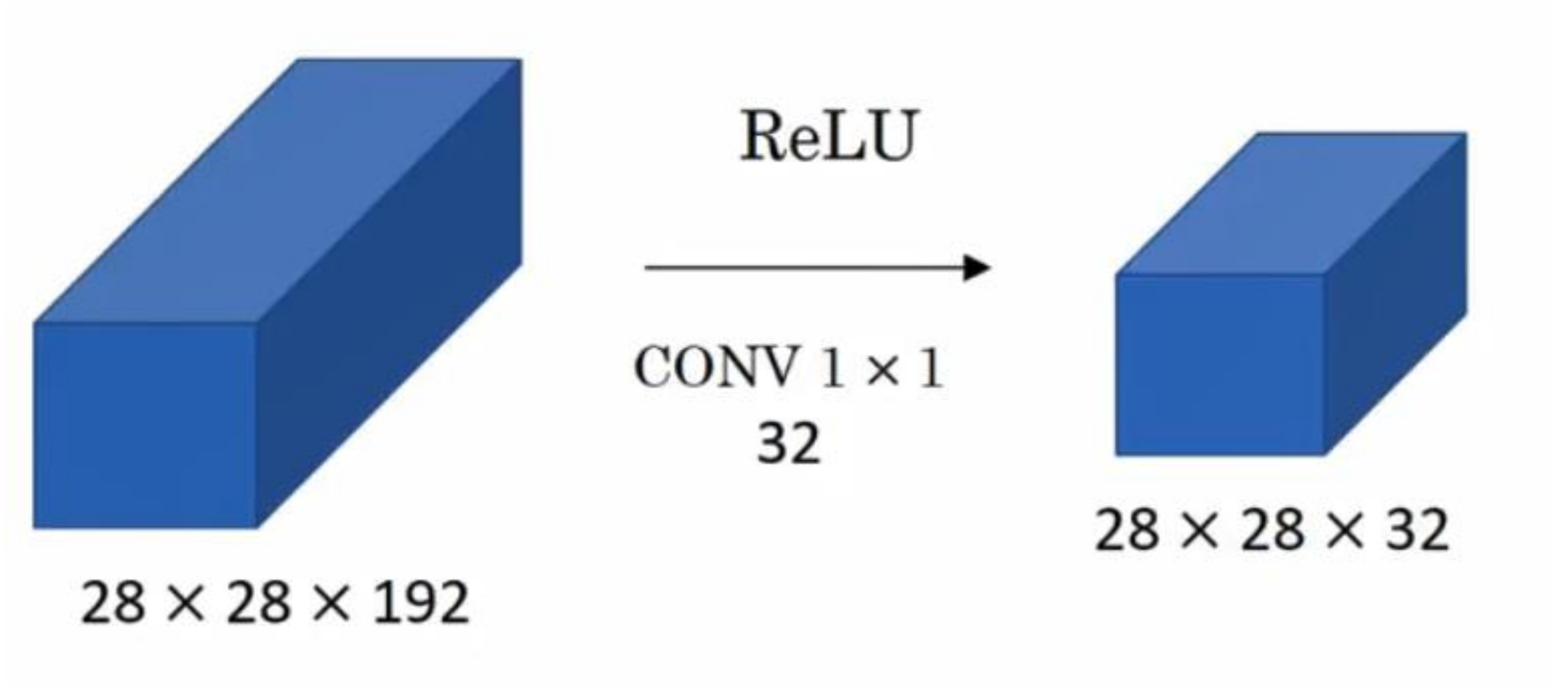
s=2   4X4 converted to 2X2

AlexNet

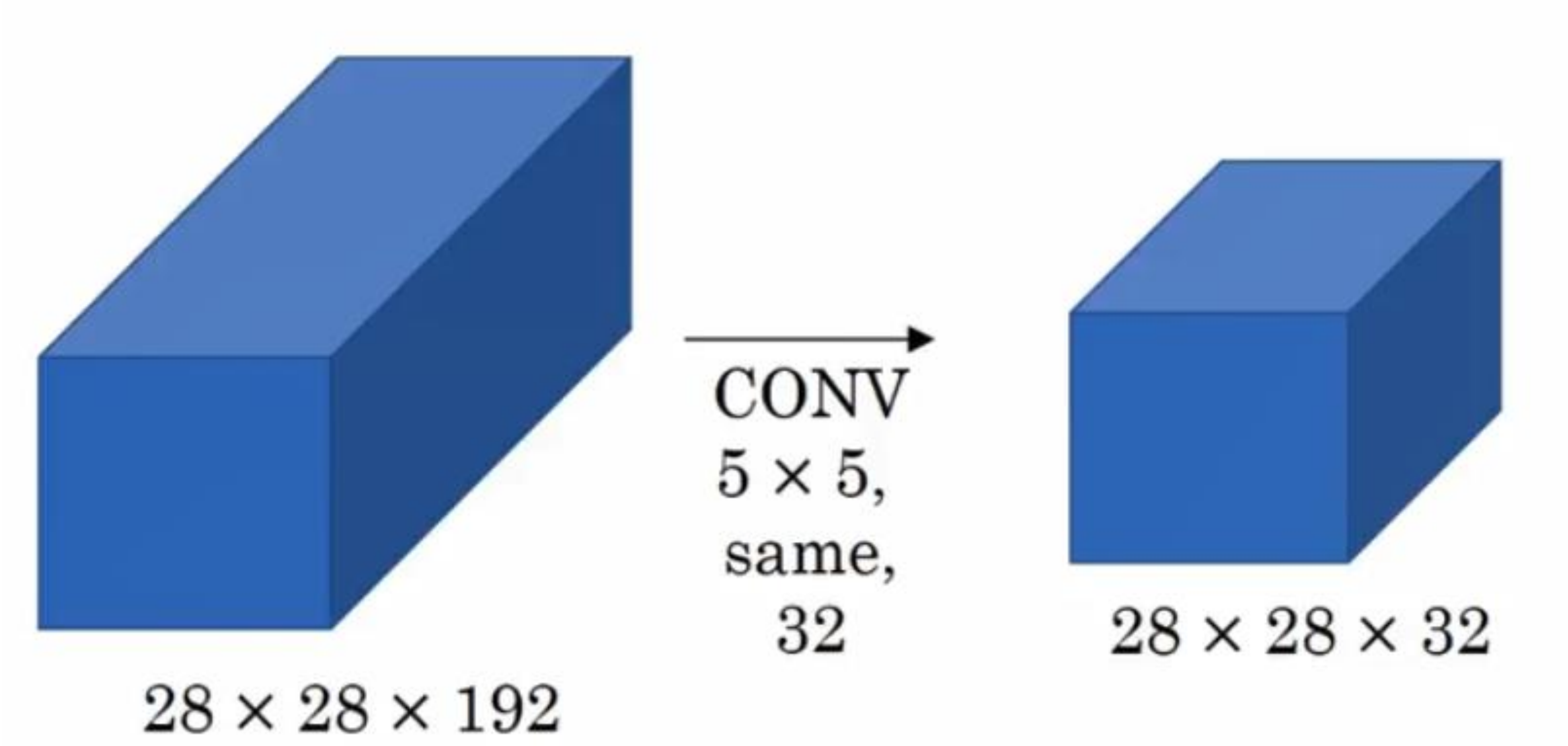


227×227 ×3
11 × 11
s = 4
55×55 × 96
MAX-POOL
3 × 3
s = 2
27×27 ×96
5 × 5
same
27×27 ×256
MAX-POOL
3 × 3
s = 2
13×13 ×256
3 × 3
same
13×13 ×384
3 × 3
13×13 ×384
3 × 3
13×13 ×256
MAX-POOL
3 × 3
s = 2
6×6 ×256
=
9216
4096
4096
Softmax
1000

Shrinking of Channels

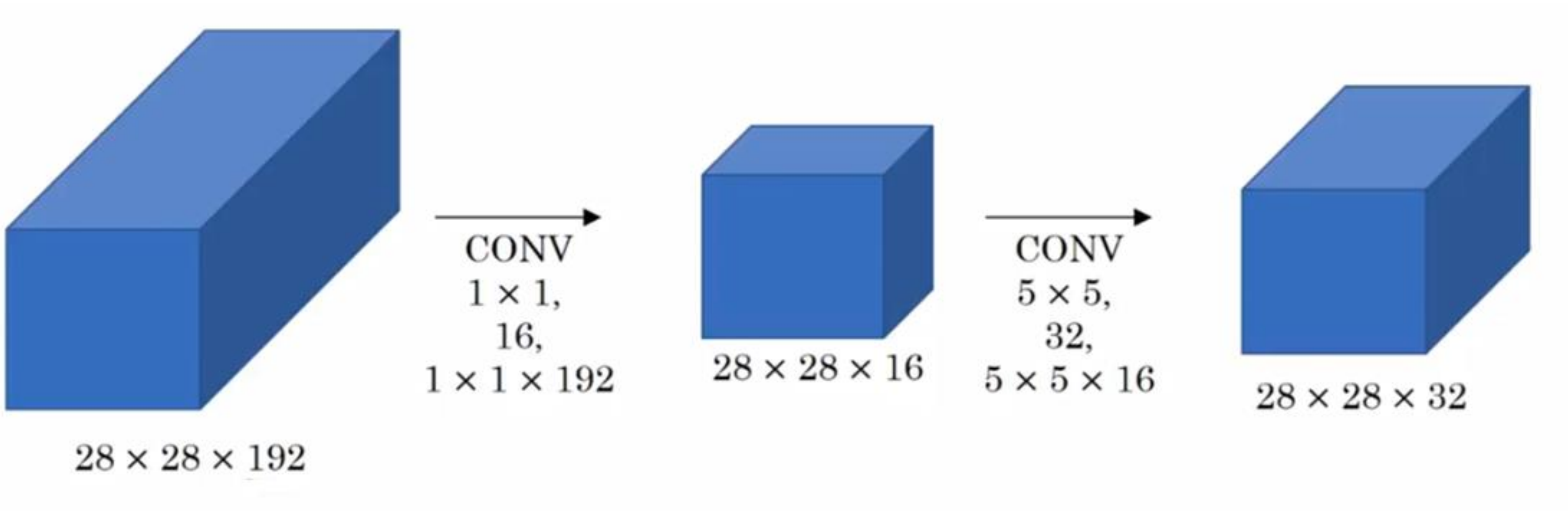


Problem of Computational Cost



28X28X32X5X5X192=120 Million Calculations

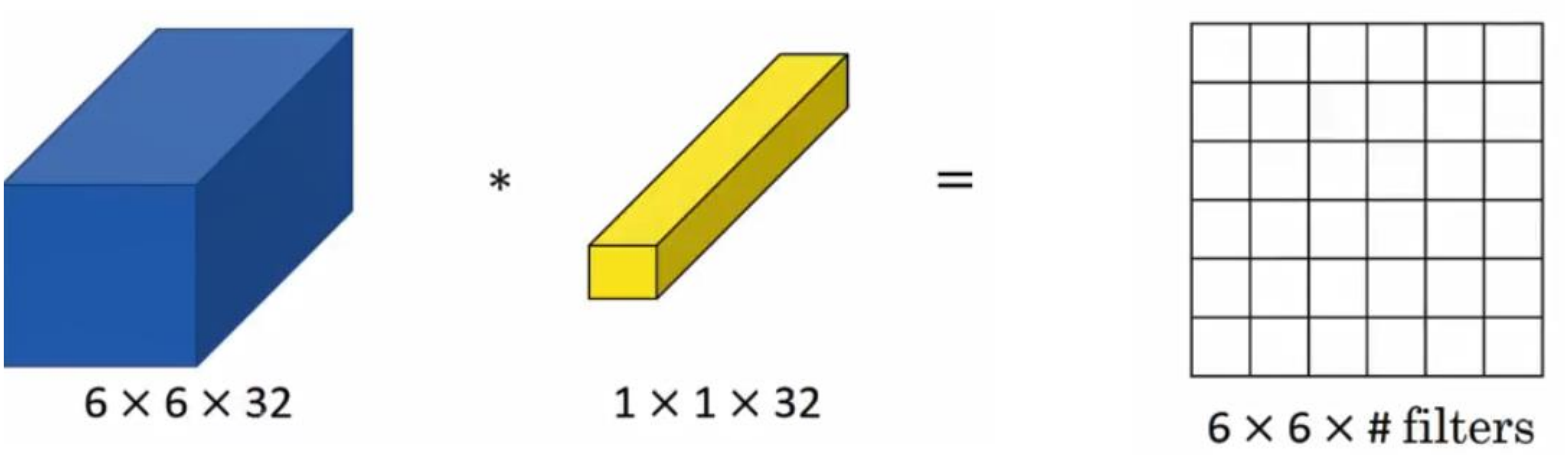
Using 1X1 Convolution to reduce the cost

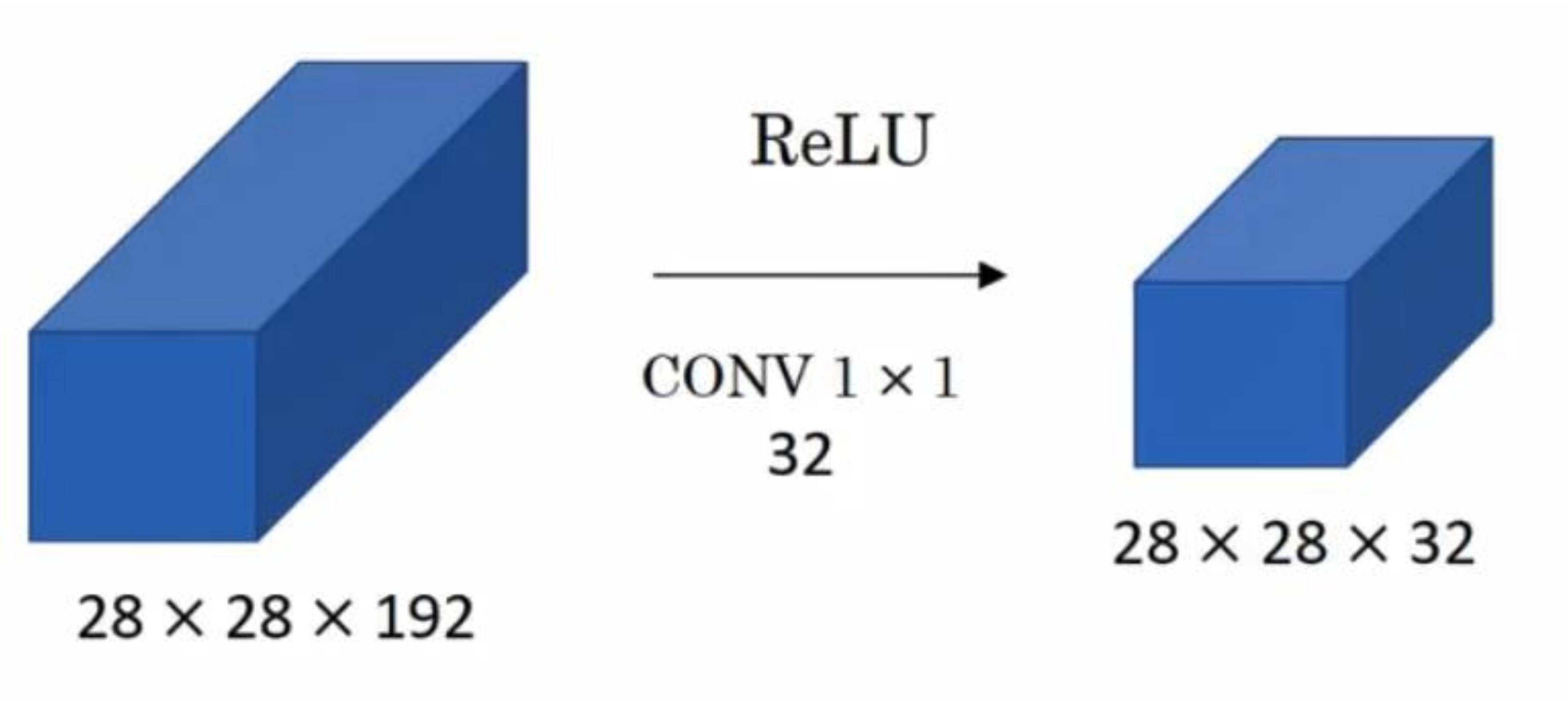


28X28X16X192 + 28X28X32X5X5X16 = 12.4 Million Calculations

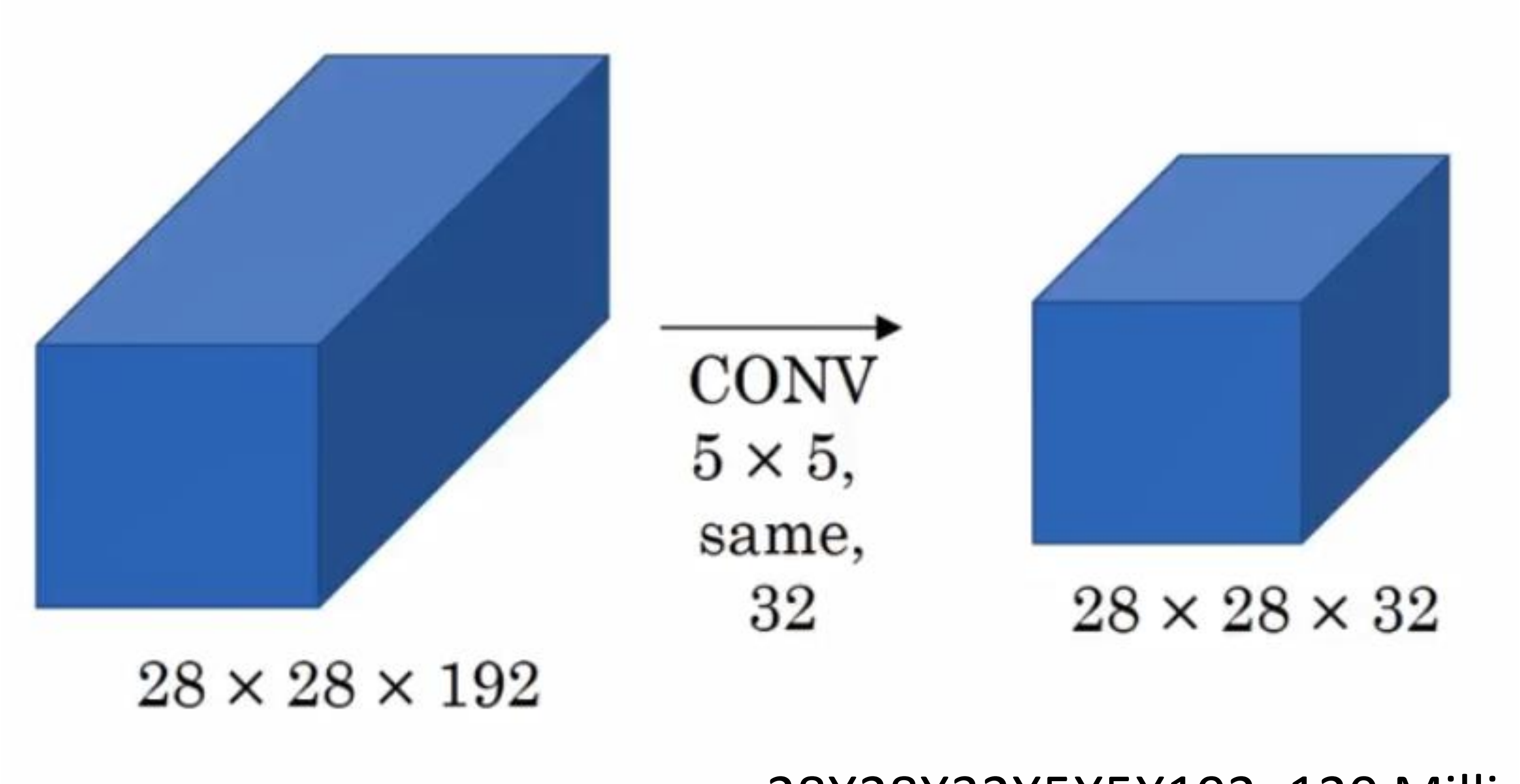
Well-known Network Architectures

Innovative use of 1X1 Convolution Filter



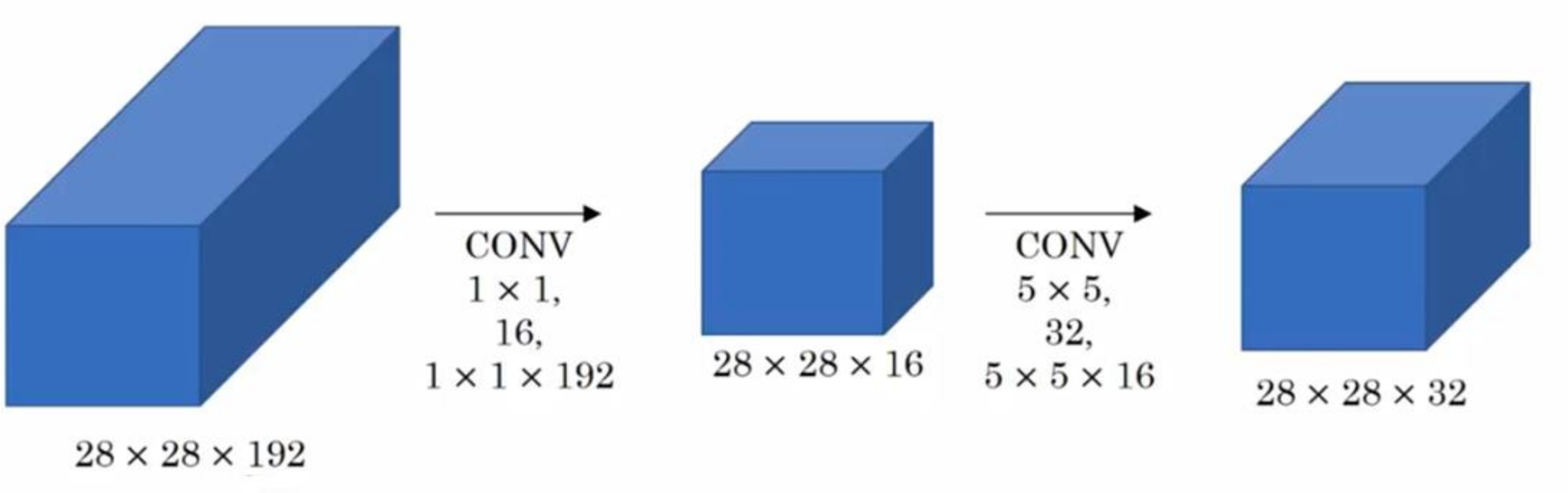

ReLU
CONV 1 × 1
32
28 × 28 × 192
28 × 28 × 32

Fire Layer



28X28X32X5X5X192=120 Million Calculations
32X5X5X192 = 153600 parameters

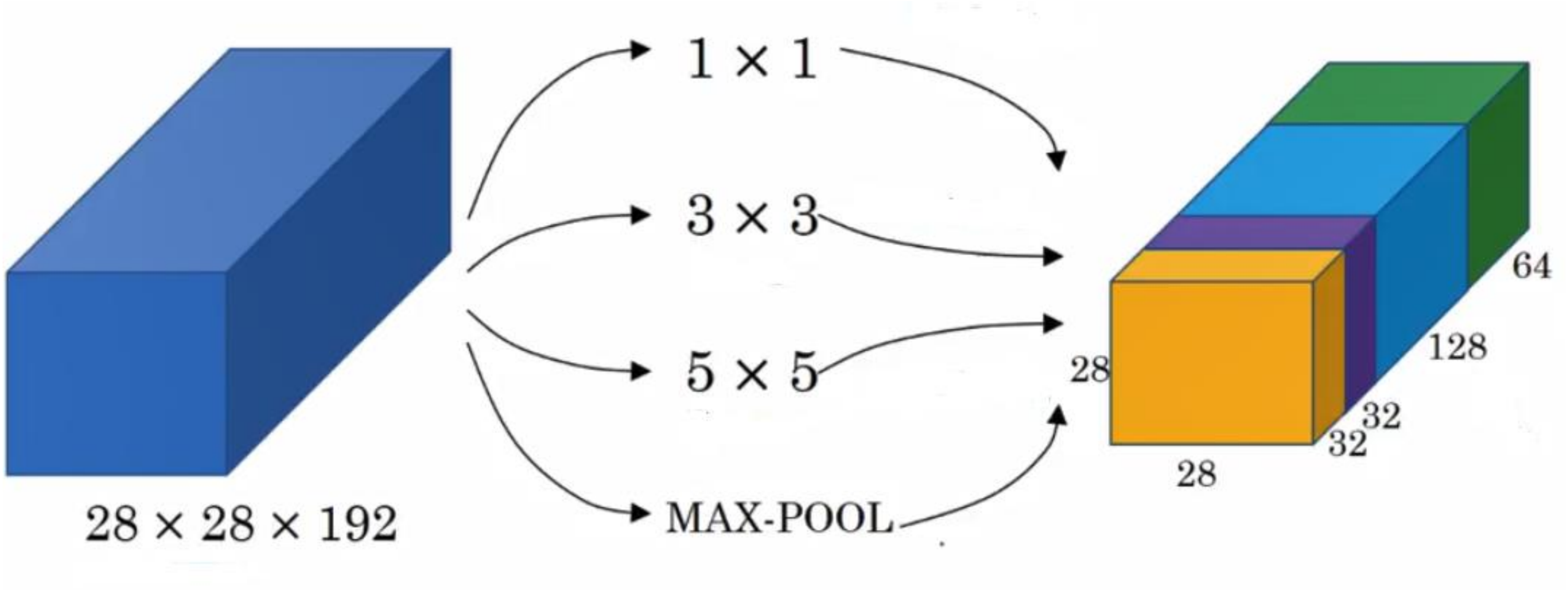
Fire Layer



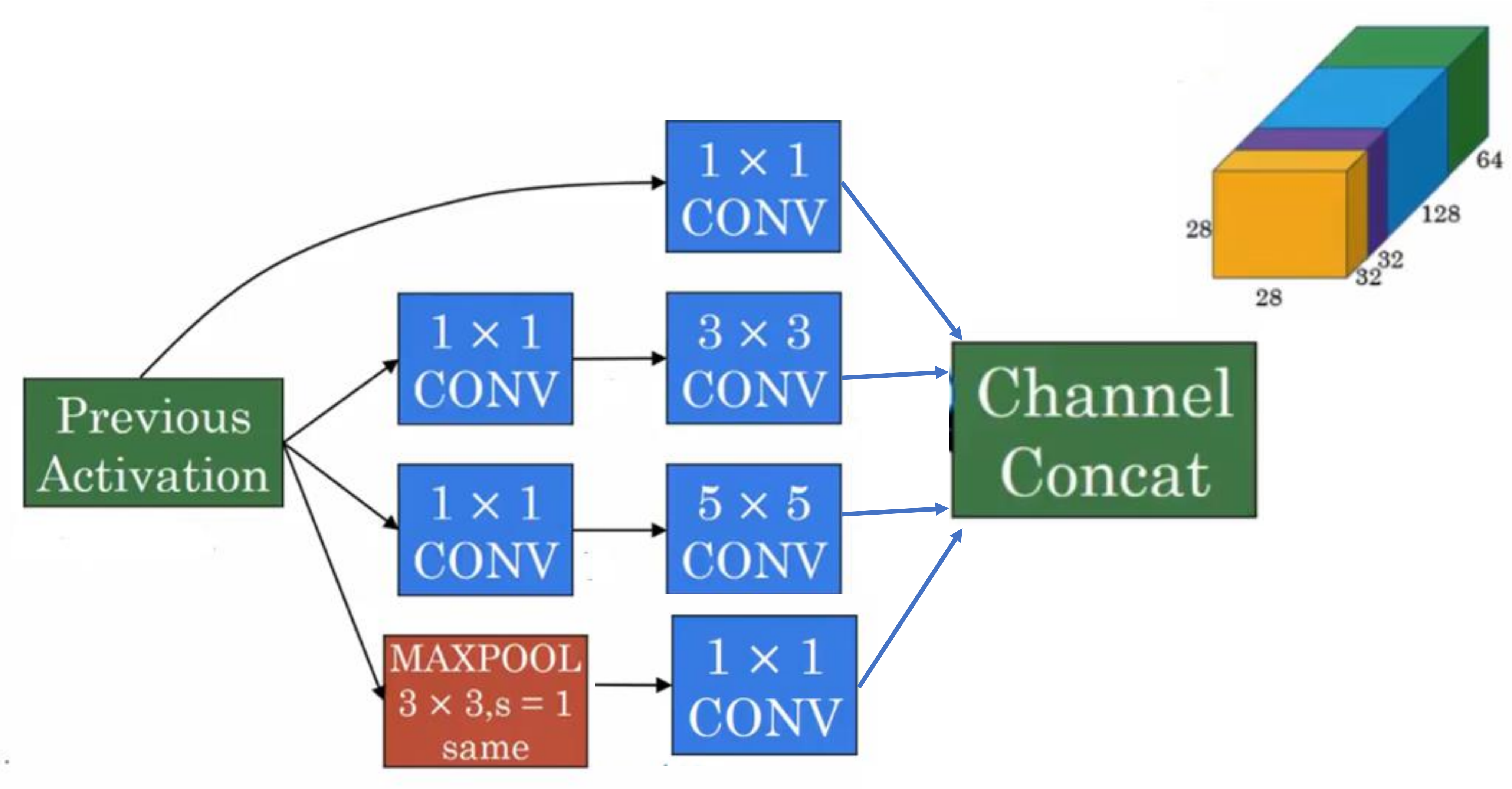
28X28X16X192 + 28X28X32X5X5X16 = 12.4 Million Calculations

16X192 + 32X5X5X16 = 15873 parameters 10x less parameters

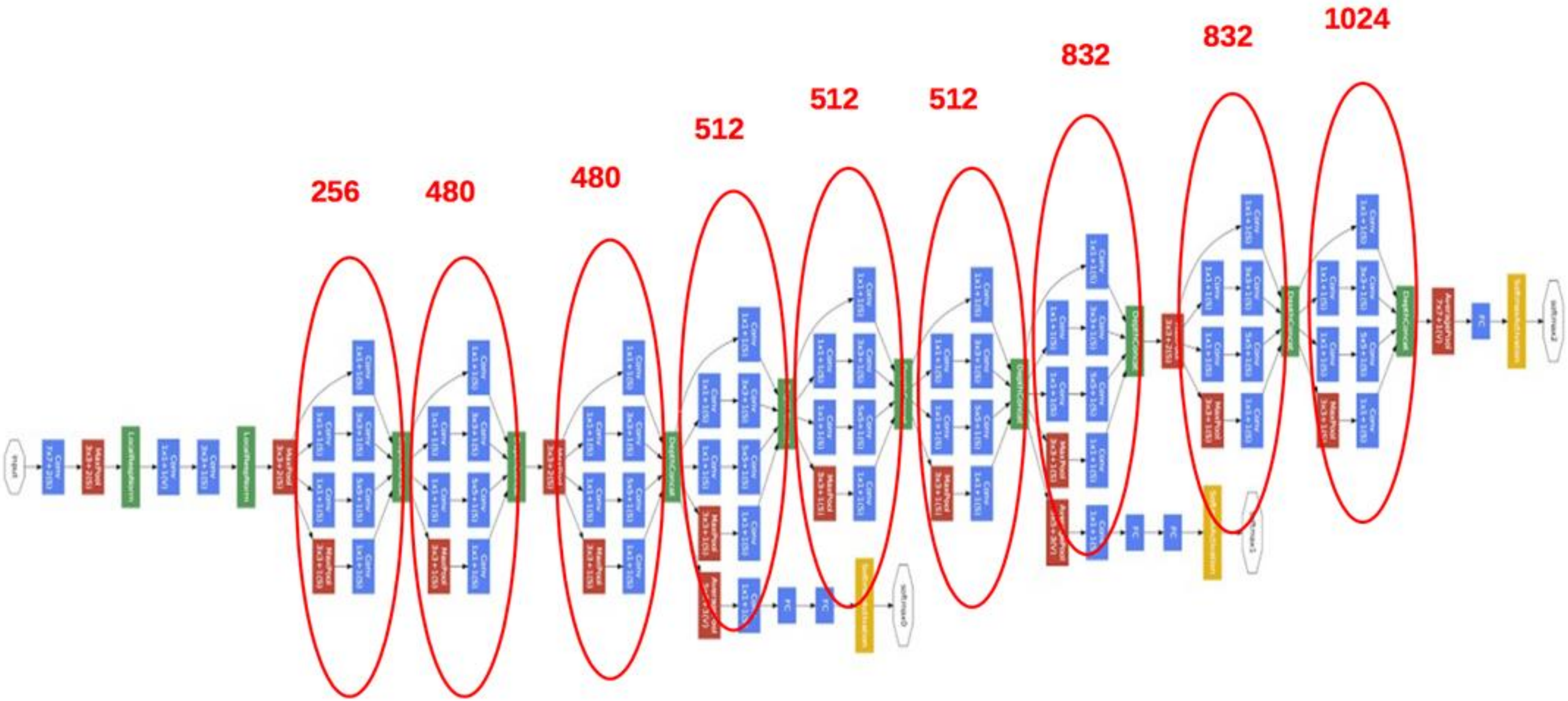
Inception Network



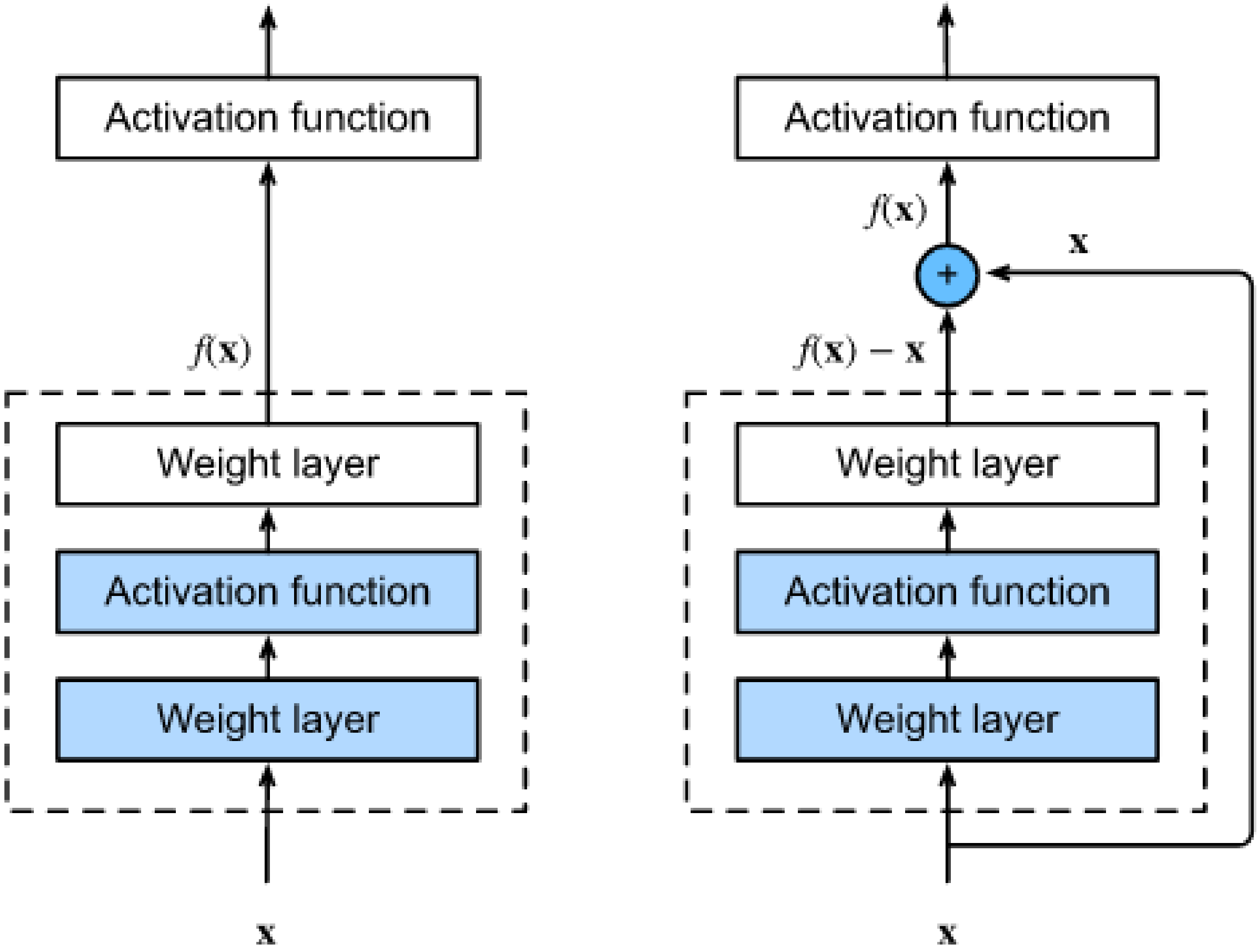
Single Unit of Inception Network



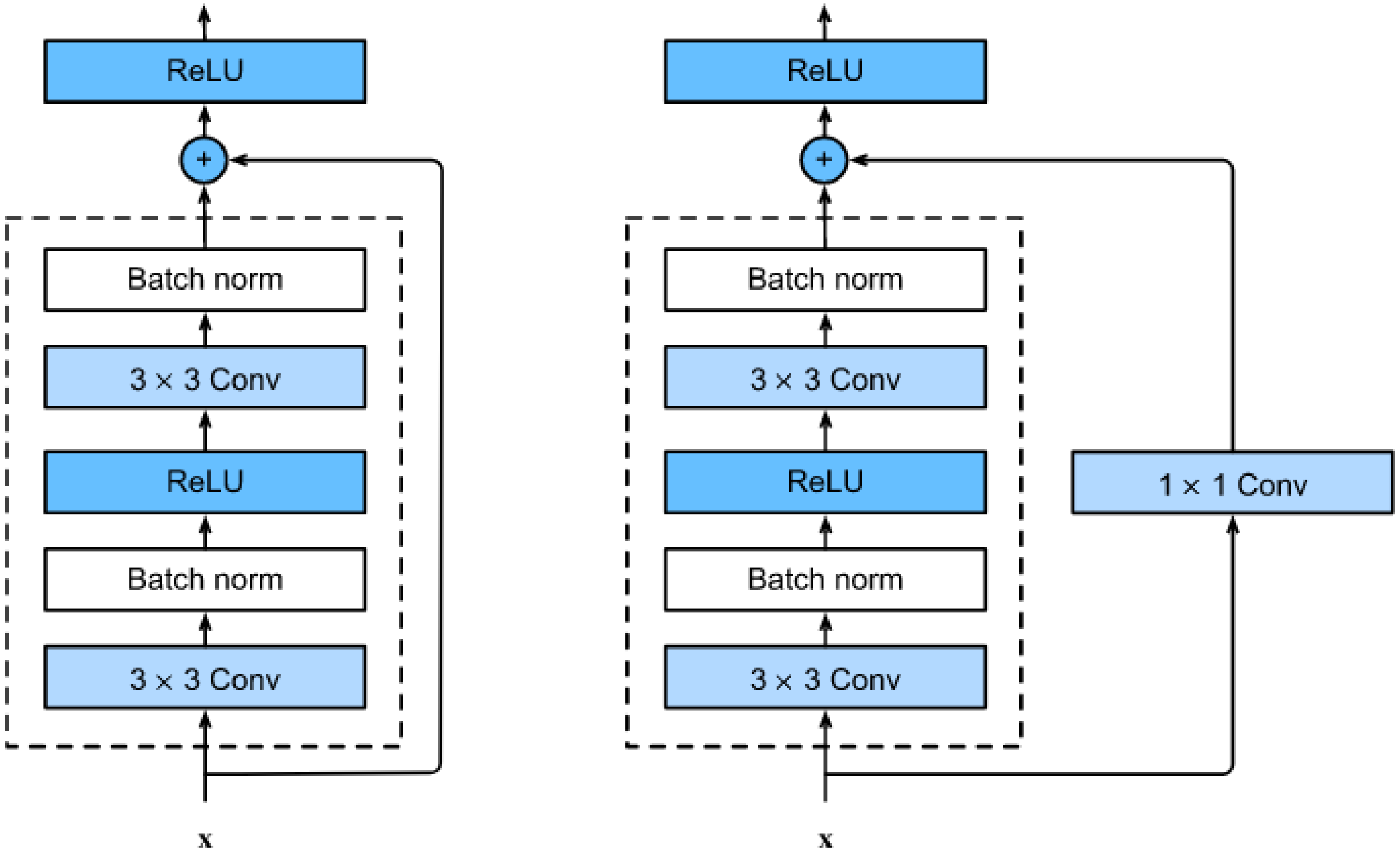
Inception Network and Inception Block



Residual Block



Residual Block



GoogLe Net: Global Average Pooling

- **Problems with fully connected (FC) layers:**

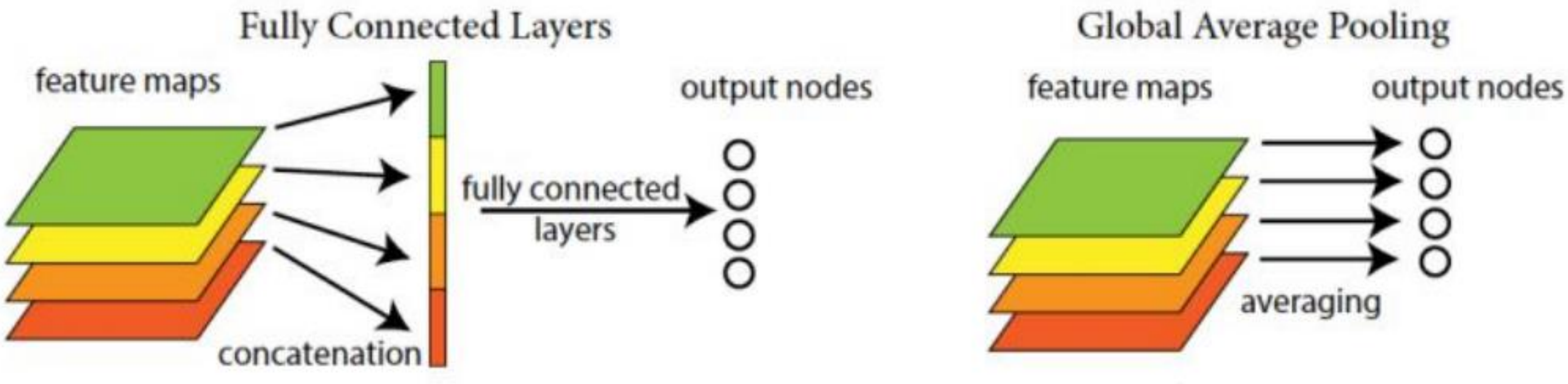
- More than 90% parameters of Alexnet and VGG are in the Fully Connected layers.

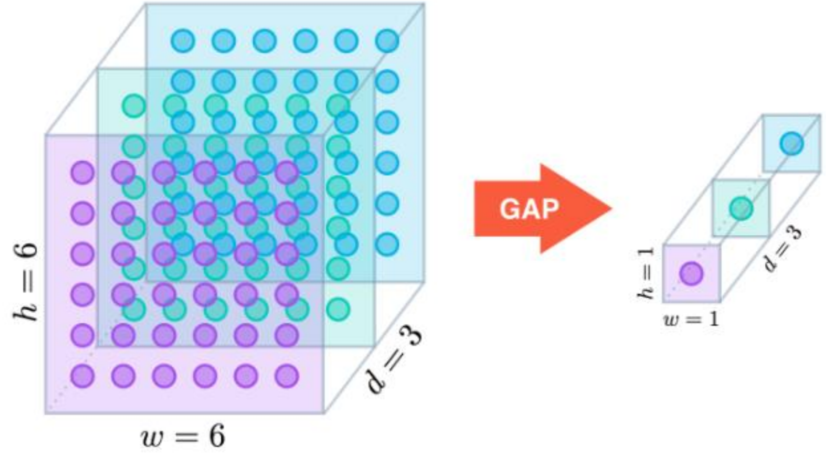
- One single particular layer in VGG contains 100 million parameters alone.

- Prone to overfitting.

- Heavily dependent on regularization methods like dropout.

GoogLe Net: Global Average Pooling





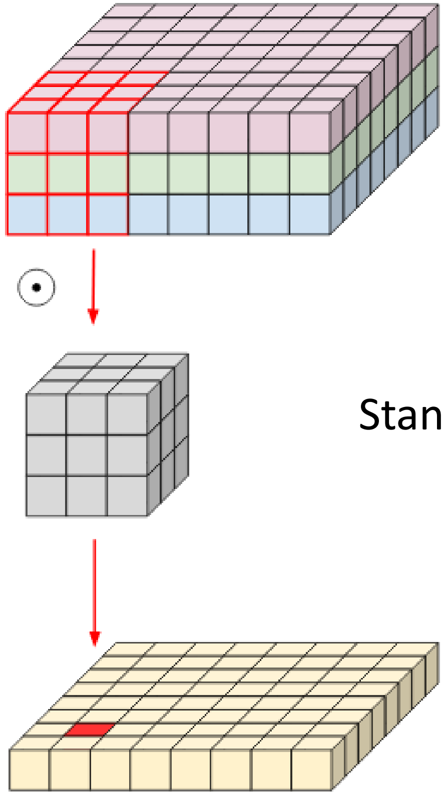
Source: https://alexisbcook.github.io/2017/global-average-pooling-layers-for-object-localization/

GoogLe Net: Global Average Pooling

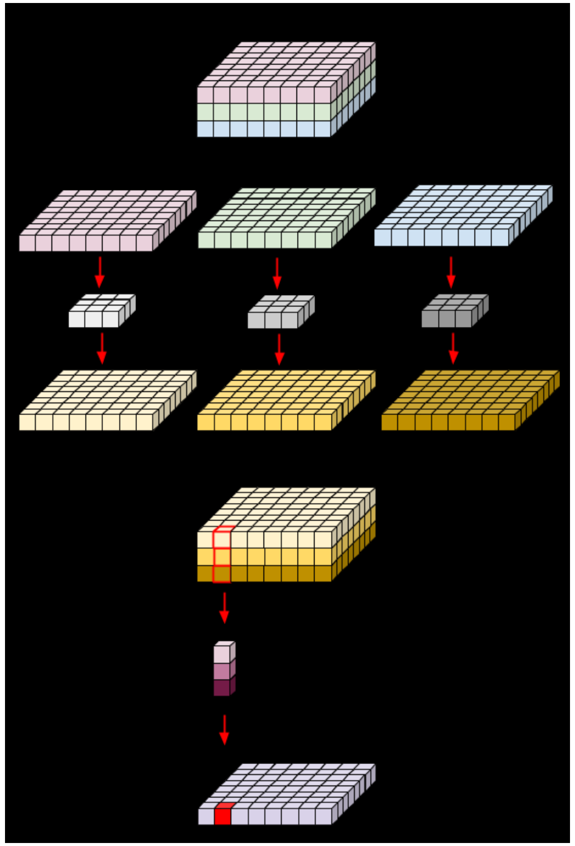
Global Average Pooling as replacement to FC layers:

- An alternative is to use spatial average of feature maps.
- Huge reduction the number of parameters as compared to the Fully Connected layer.
- Stronger local modelling using the micro network.
- It is itself a structural regularizer and hence doesn’t need dropout.

Modified Convolution Operation



Standard Convolution

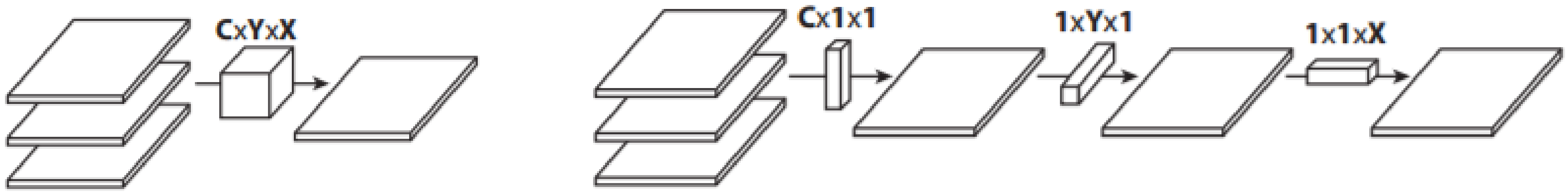


Depth-wise Convolution

Point-wise Convolution

Flattened Convolutions: Fire Layer

- Replace $c \times y \times x$ convolutions with $c \times 1 \times 1$, $1 \times y \times 1$, and $1 \times 1 \times x$ convolutions



3D convolution

1D convolution over different direction

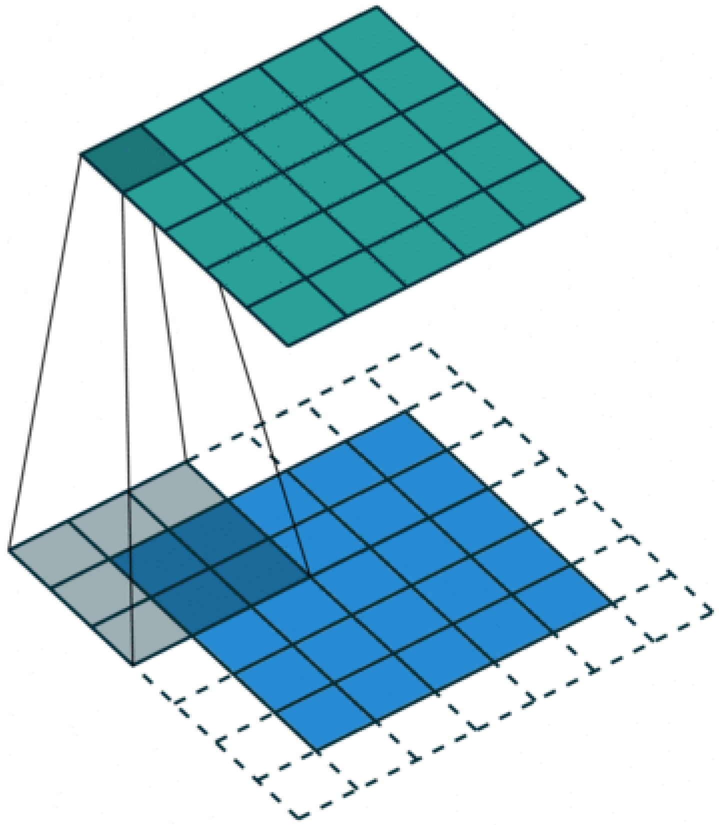
Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size."

2D Convolution

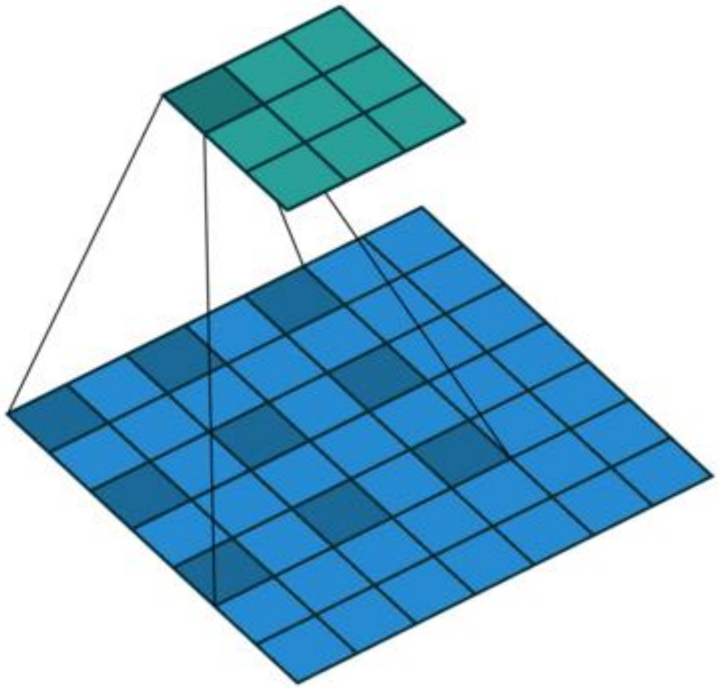
•**Kernel Size**: The kernel size defines the field of view of the convolution. A common choice for 2D is 3 i.e 3x3 pixels.

•**Stride**: The stride defines the step size of the kernel when traversing the image. While its default is usually 1, we can use a stride of 2 for downsampling an image similar to MaxPooling.

•**Padding**: The padding defines how the border of a sample is handled. A (half) padded convolution will keep the spatial output dimensions equal to the input, whereas unpadded convolutions will crop away some of the borders if the kernel is larger than 1.



Dilated Convolution

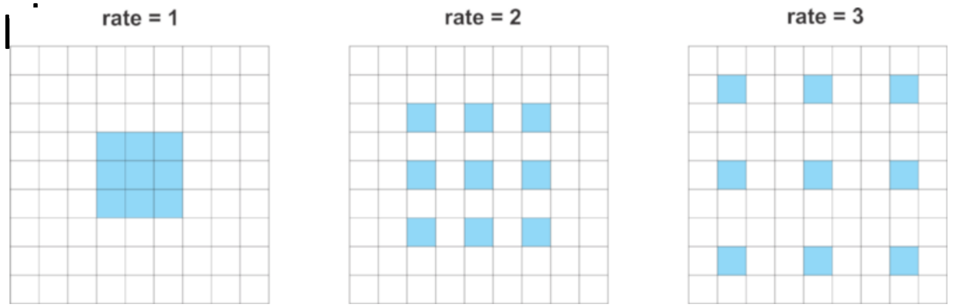


- Dilated convolutions are particularly popular in the field of real-time segmentation.

- Dilated convolutions introduce another parameter to convolutional layers called the **dilation rate,** that defines a spacing between the values in a kernel.

- A 3x3 kernel with a dilation rate of 2 will have the same field of view as a 5x5 kernel, while only using 9 parameters. Imagine taking a 5x5 kernel and deleting every second column and row.

- Use them if you need a wide field of view and cannot afford multi convolutions or larger kernels.



Dilated Convolution

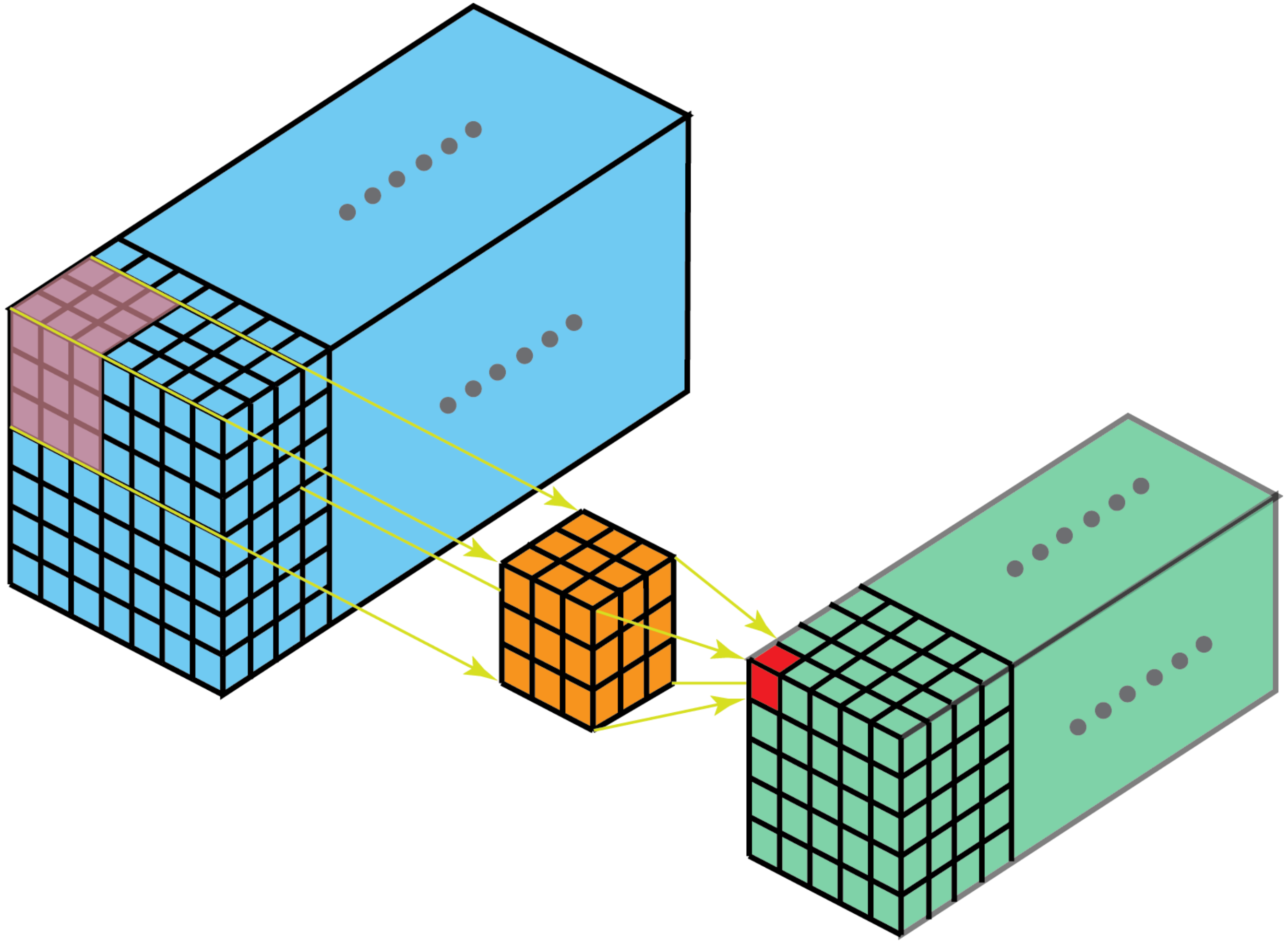
- Dilated convolutions have generally improved performance in semantic segmentation results

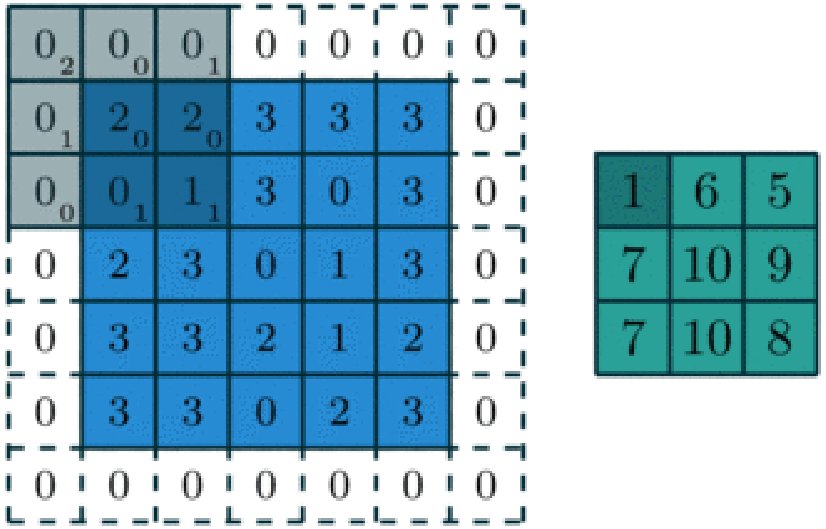
- The architecture is based on the fact that dilated convolutions support exponential expansion of the receptive field without loss of resolution or coverage.

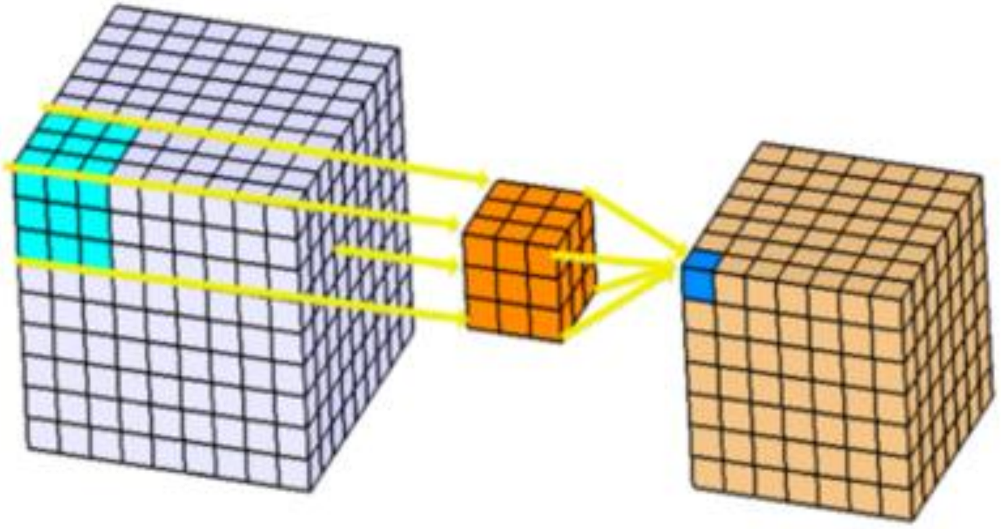
- Allows one to have larger receptive field with same computation and memory costs while also preserving resolution.

- Pooling and Strided Convolutions are similar concepts but both reduce the resolution.

3D convolution





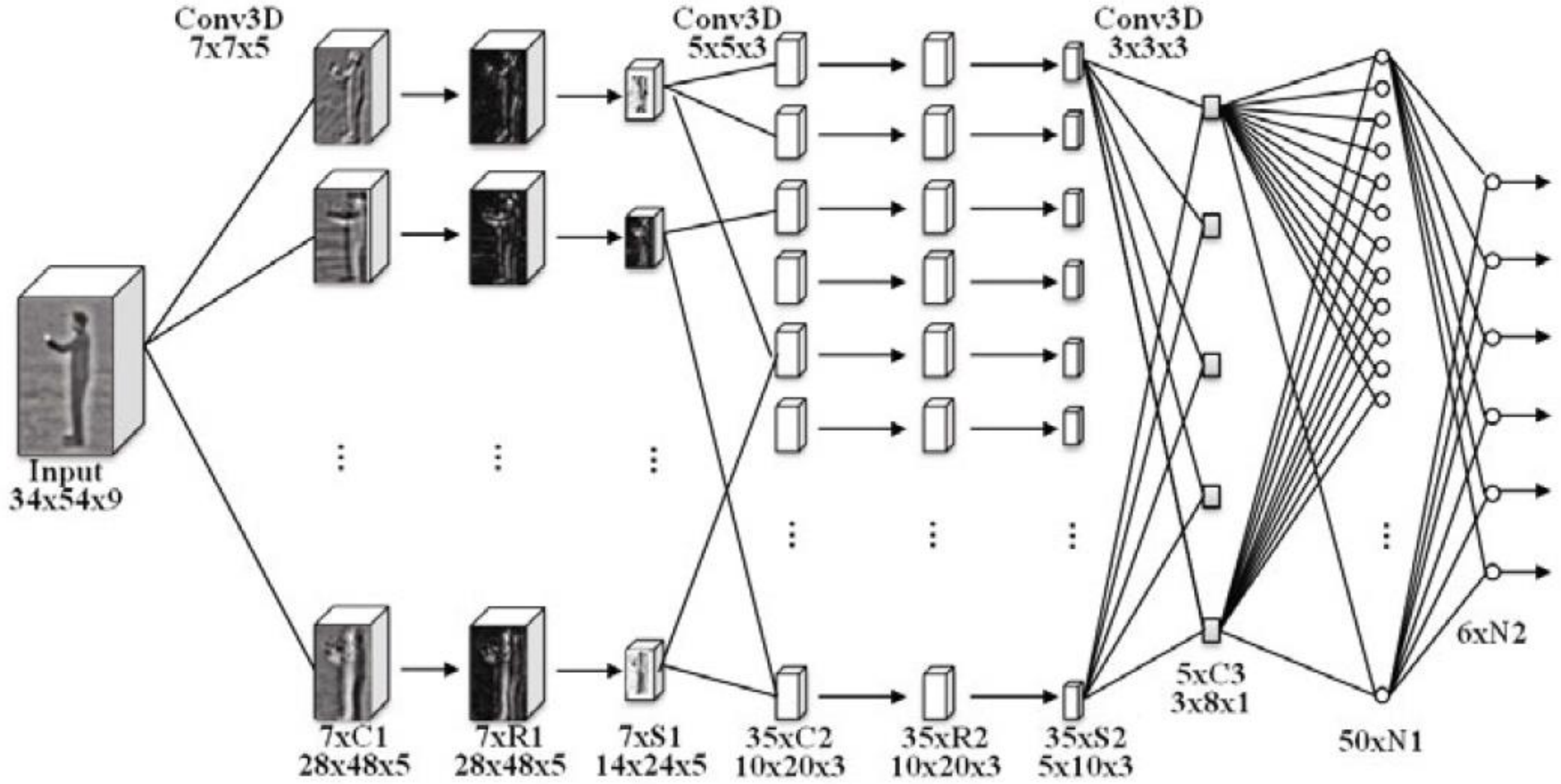


2D vs 3D Convolution

3D CNN

- 3D imageries such as MRI Scan, Human action in video sequence should be processed frame-by-frame
- Temporal property within the 3D imagery is important in extracting the features
- 2D convolution extracts features from only spatial dimensions
- 3D convolution operates on the input volume not only in the *X* and *Y* dimensions but also in *Z* dimension
- Building a 3D CNN requires, 3D Convolution as well as 3D pooling

3D CNN for Action Recognition



Transfer Learning

- Sharing the Knowledge gained solving one problem and applying to a different but related problem

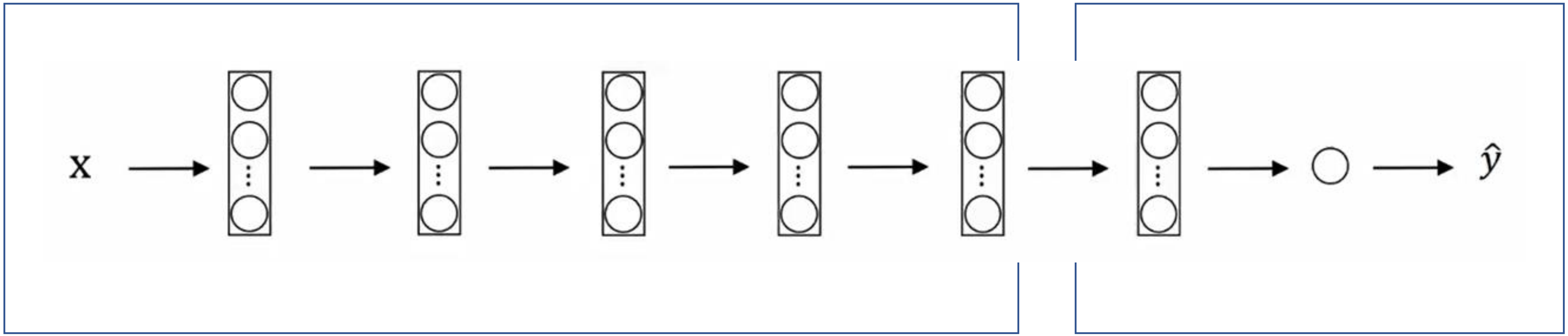
- Transfer Learning is next popular driver of deep learning after supervised learning

- The feature spaces of the source and target domain are different, e.g. the documents are written in two different languages.

- The marginal probability distributions of source and target domain are different, e.g. the documents discuss different topics.

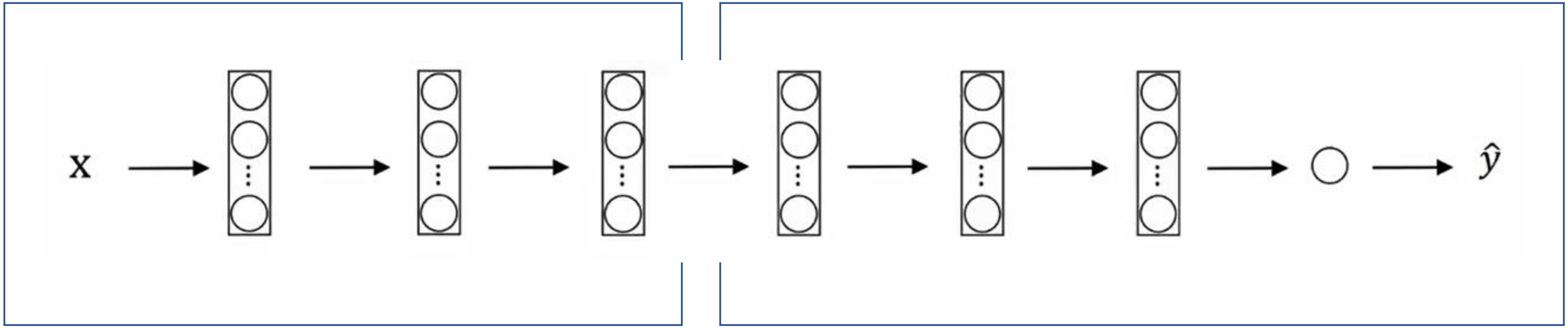
- Simulation training is becoming a hot area within the sphere of Deep Learning. Few labs have also started using AR/VR Technologies to be integrated for making advance learning models for some of the critical problems area

Transfer Learning – Small Data



- Finding a Bengal Cat or British Cat where you have small dataset of these categories of cats
- You can download a cat classifier and train last few layers on your data specifically
- In Popular Deep Learning platforms, now you have good support for transfer learning
- Functions like Trainable_Parameters and Freezing specific layers are available

Transfer Learning-Mid Size Data

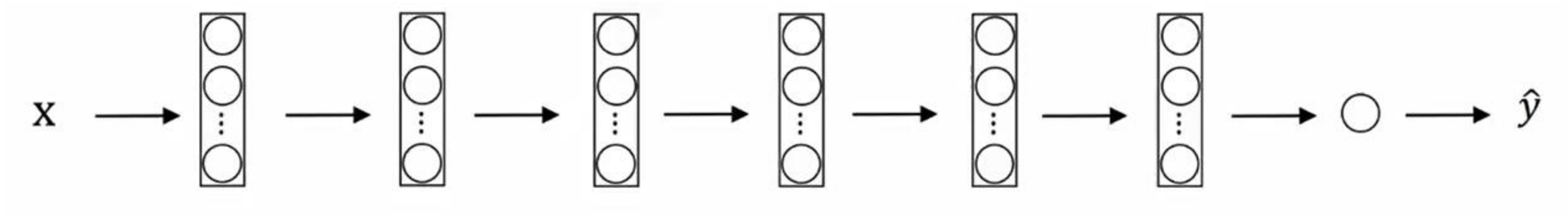


- In this category we use initial set of layers from the open source model and use the trained weight values.

- Remaining layers there are two ways to handle:

    - we can train the layers from start
    - we can start the weight optimization from the existing set of weights that we have received from the open source model

Transfer Learning- Enough Data



- Where we have enough data to train our new model, open source model may still be useful.

- We can use the pre-trained weights of the model from the same domain and consider that as a starting point for our model.

- It may be much easier and faster to adapt these model for new set of data that we want to train upon.

Face Identification Vs. Verification

- Verification
  - Input an image and also the Name/ID of the person
  - Output whether the input image is of the claimed person

- Identification
  - Input an Image
  - Output whether the image is any of the K persons in the database

Dimension and Parameters

Output dimension and number of parameters

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)			96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)			3x3 pool, stride 2
Conv2	Convolution (5x5)			256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)			3x3 pool, stride 2
Conv3	Convolution (3x3)			384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

Output dimension and number of parameters **(Conv1)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54 , 54 , 96)	11 * 11 * 3 * 96 + 96	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26 , 26 , 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(25 , 25 , 256)	5 * 5 * 96 * 256 + 256	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12 , 12 , 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12 , 12 , 384)	3 * 3 * 256 * 384 + 384	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$No.of\ Parameters$=(Filter_Height × Filter_Width × No. of Input Channels) × No. of Filters + Number of Filters

$$No.of\ Parameters = (11 * 11 * 3 * 96) + 96 = 34{,}944$$

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(224 - 11 + 2*0)}{4} + 1 = \frac{213}{4} + 1 = 54$$

Output dimension and number of parameters **(Pool1)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)			3x3 pool, stride 2
Conv2	Convolution (5x5)			256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)			3x3 pool, stride 2
Conv3	Convolution (3x3)			384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(54 - 3 + 2 * 0)}{2} + 1 = 26$$

Output dimension and number of parameters **(Conv2)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)			256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)			3x3 pool, stride 2
Conv3	Convolution (3x3)			384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$No.of\ Parameters$=(Filter_Height × Filter_Width × No. of Input Channels) × No. of Filters + Number of Filters

$$No.of\ Parameters = (5 * 5 * 96 * 256) + 256 = 6{,}14{,}656$$

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(26 - 5 + 2 * 2)}{1} + 1 = 26$$

Output dimension and number of parameters **(Pool2)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)			3x3 pool, stride 2
Conv3	Convolution (3x3)			384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2*Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(26 - 3 + 2*0)}{2} + 1 = 12$$

Output dimension and number of parameters **(Conv3)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)			384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$No.of\ Parameters$=(Filter_Height × Filter_Width × No. of Input Channels) × No. of Filters + Number of Filters

$$No.of\ Parameters = (3 * 3 * 256 * 384) + 384 = 8{,}85{,}120$$

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(12 - 3 + 2 * 1)}{1} + 1 = 12$$

Output dimension and number of parameters **(Conv4)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)			384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$No.of\ Parameters$=(Filter_Height × Filter_Width × No. of Input Channels) × No. of Filters + Number of Filters

$$No.of\ Parameters = (3 * 3 * 384 * 384) + 384 = 13{,}27{,}488$$

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(12 - 3 + 2 * 1)}{1} + 1 = 12$$

Output dimension and number of parameters **(Conv5)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	13,27,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)			256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$No.of\ Parameters$=(Filter_Height × Filter_Width × No. of Input Channels) × No. of Filters + Number of Filters

$$No.of\ Parameters = (3 * 3 * 384 * 256) + 256 = 8{,}84{,}992$$

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(12 - 3 + 2 * 1)}{1} + 1 = 12$$

Output dimension and number of parameters **(Pool3)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	13,27,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)	(12, 12, 256)	8,84,992	256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)			3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$$Output\ Size = \frac{(Input\ Size - Filter\ Size + 2 * Padding)}{Stride} + 1$$

$$Output\ Size = \frac{(12 - 3 + 2 * 0)}{2} + 1 = 5$$

Output dimension and number of parameters **(Flatten)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	13,27,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)	(12, 12, 256)	8,84,992	256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)	(5, 5, 256)	0	3x3 pool, stride 2
Flatten	-			Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$$\boxed{Flatten = Height * Width * Channel = 5 * 5 * 256 = 6400}$$

Output dimension and number of parameters **(FC1)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	13,27,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)	(12, 12, 256)	8,84,992	256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)	(5, 5, 256)	0	3x3 pool, stride 2
Flatten	-	(6400 X 1)	0	Flatten to 1D
FC1	Fully Connected			4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$$\boxed{No.of\ Parameters = (6400 * 4096) + 4096 = 2,62,14,656}$$

Output dimension and number of parameters **(FC2)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	13,27,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)	(12, 12, 256)	8,84,992	256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)	(5, 5, 256)	0	3x3 pool, stride 2
Flatten	-	(6400 X 1)	0	Flatten to 1D
FC1	Fully Connected	4096	2,62,14,656	4096 units
FC2	Fully Connected			4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$$\boxed{No.of\ Parameters = (4096 * 4096) + 4096 = 1{,}67{,}81{,}312}$$

Output dimension and number of parameters **(FC3)**

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	6,14,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	8,85,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	13,27,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)	(12, 12, 256)	8,84,992	256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)	(5, 5, 256)	0	3x3 pool, stride 2
Flatten	-	(6400 X 1)	0	Flatten to 1D
FC1	Fully Connected	4096	2,62,14,656	4096 units
FC2	Fully Connected	4096	1,67,81,312	4096 units
FC3 (Output)	Fully Connected			1000 units for classification

$No. of\ Parameters$=($4096 * 1000$) + 1000 = 40,97,000

Output dimension and number of parameters

Layer	Type	Output Shape	Number of Parameters	Explanation
Input	-	(224, 224, 3)	0	Input RGB image
Conv1	Convolution (11x11)	(54, 54, 96)	(11*11*3*96) + 96 = 34,944	96 filters, stride 4, padding 0
Pool1	Max Pooling (3x3)	(26, 26, 96)	0	3x3 pool, stride 2
Conv2	Convolution (5x5)	(26, 26, 256)	(5*5*96*256) + 256 = 614,656	256 filters, stride 1, padding 2
Pool2	Max Pooling (3x3)	(12, 12, 256)	0	3x3 pool, stride 2
Conv3	Convolution (3x3)	(12, 12, 384)	(3*3*256*384) + 384 = 885,120	384 filters, stride 1, padding 1
Conv4	Convolution (3x3)	(12, 12, 384)	(3*3*384*384) + 384 = 1,327,488	384 filters, stride 1, padding 1
Conv5	Convolution (3x3)	(12, 12, 256)	(3*3*384*256) + 256 = 884,992	256 filters, stride 1, padding 1
Pool3	Max Pooling (3x3)	(5, 5, 256)	0	3x3 pool, stride 2
Flatten	-	6400	0	Flatten to 1D
FC1	Fully Connected	4096	(6400*4096) + 4096 = 26,214,656	4096 units
FC2	Fully Connected	4096	(4096*4096) + 4096 = 16,781,312	4096 units
FC3 (Output)	Fully Connected	1000	(4096*1000) + 1000 = 4,097,000	1000 units for classification

Backpropogation CNN

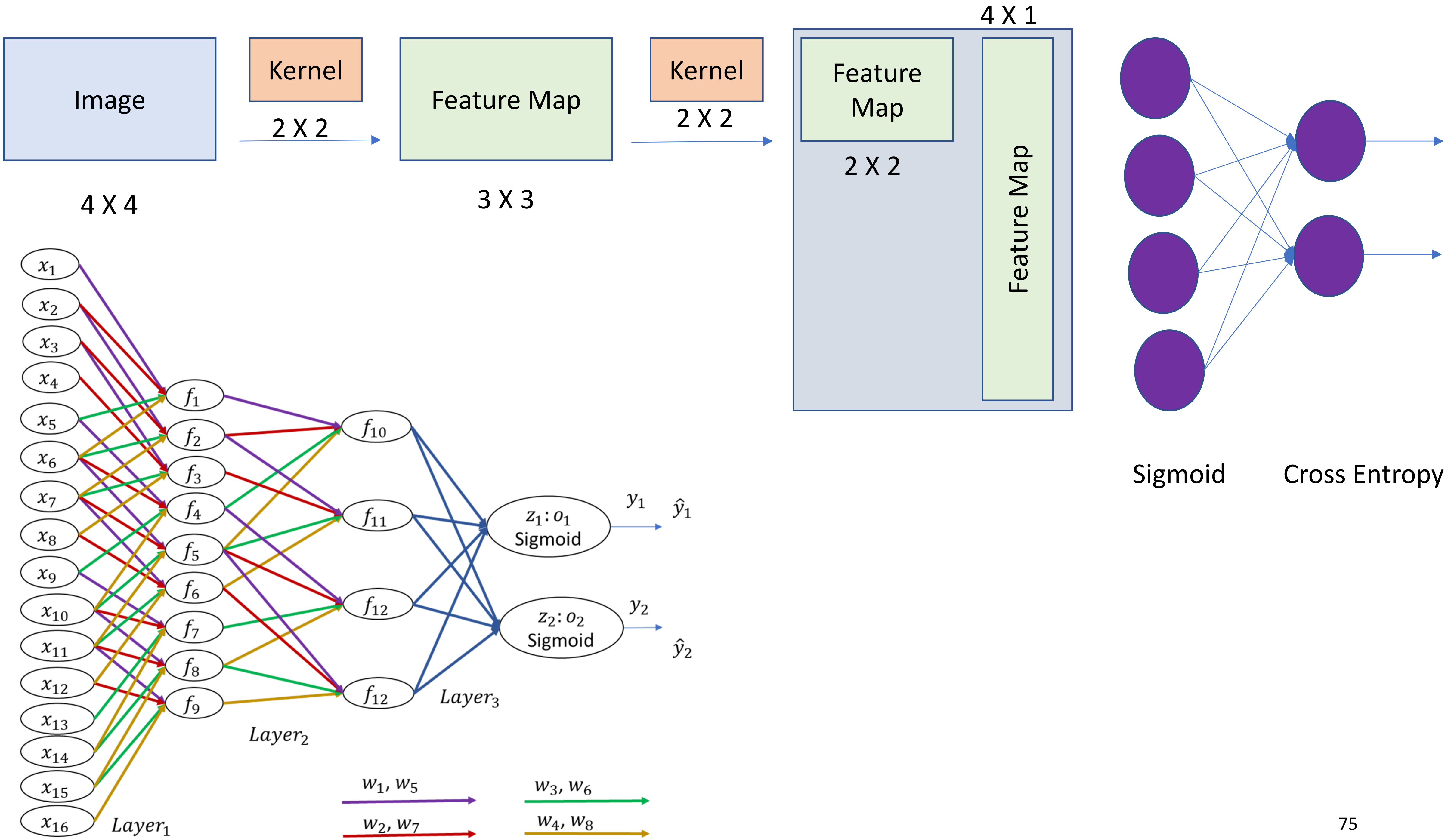


Image
4 X 4
Kernel
2 X 2
Feature Map
3 X 3
Kernel
2 X 2
Feature Map
2 X 2
4 X 1
Feature Map
Sigmoid
Cross Entropy
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
x_9
x_{10}
x_{11}
x_{12}
x_{13}
x_{14}
x_{15}
x_{16}
$Layer_1$
f_1
f_2
f_3
f_4
f_5
f_6
f_7
f_8
f_9
$Layer_2$
f_{10}
f_{11}
f_{12}
f_{12}
$Layer_3$
$z_1 : o_1$ Sigmoid
$z_2 : o_2$ Sigmoid
y_1
$\hat{y}_1$
y_2
$\hat{y}_2$
w_1, w_5
w_2, w_7
w_3, w_6
w_4, w_8

Thank You