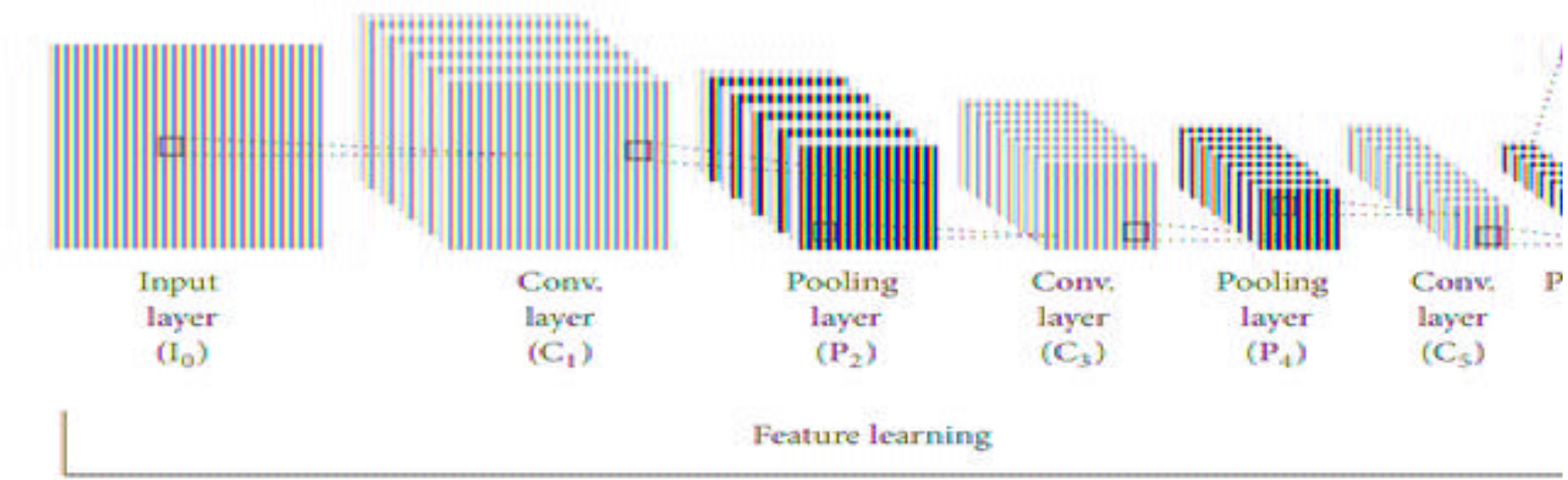
Building the first CNN Architecture & Parameters Calculations



--- Ramendra Kumar ---

Convolution Operation : Input with one channel

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

Kernel/Filter

Stride?

1x1	1x0	1x1	0	0
0x0	1x1	1x0	1	0
0x1	0x0	1x1	1	1
0	0	1	1	0
0	1	1	0	0

4		

Output

Convolution Operation : Input with three channel

0	0	0	0	0	0	...
0	156	155	156	158	158	...
0	153	154	157	159	159	...
0	149	151	155	158	159	...
0	146	146	149	153	158	...
0	145	143	143	148	158	...
...	...	...	...	...	...	...

Input Channel #1 (Red)

0	0	0	0	0	0	...
0	167	166	167	169	169	...
0	164	165	168	170	170	...
0	160	162	166	169	170	...
0	156	156	159	163	168	...
0	155	153	153	158	168	...
...	...	...	...	...	...	...

Input Channel #2 (Green)

0	0	0	0	0	0	...
0	163	162	163	165	165	...
0	160	161	164	166	166	...
0	156	158	162	165	166	...
0	155	155	158	162	167	...
0	154	152	152	157	167	...
...	...	...	...	...	...	...

Input Channel #3 (Blue)

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

Kernel Channel #1

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

Kernel Channel #2

0	1	1
0	1	0
1	-1	1

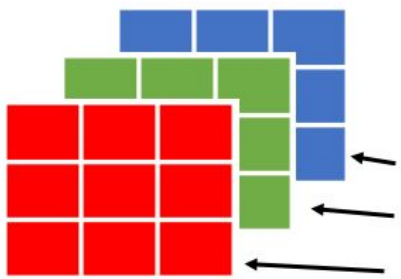
Kernel Channel #3

$+ \quad -498 \quad + \quad 164 \quad + 1 = -25$

Bias = 1

Output

-25				...
				...
				...
				...
...	...	...	...	...

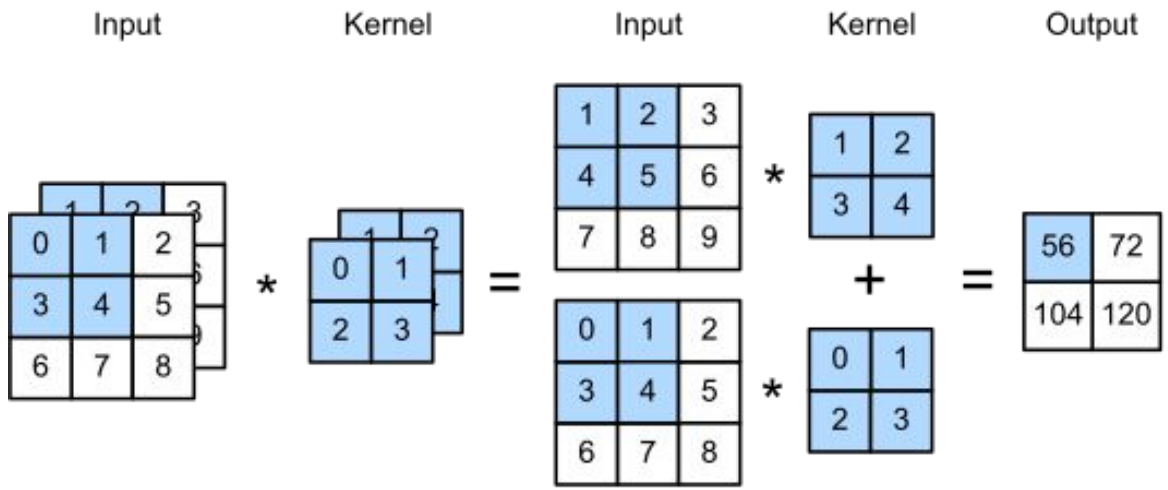


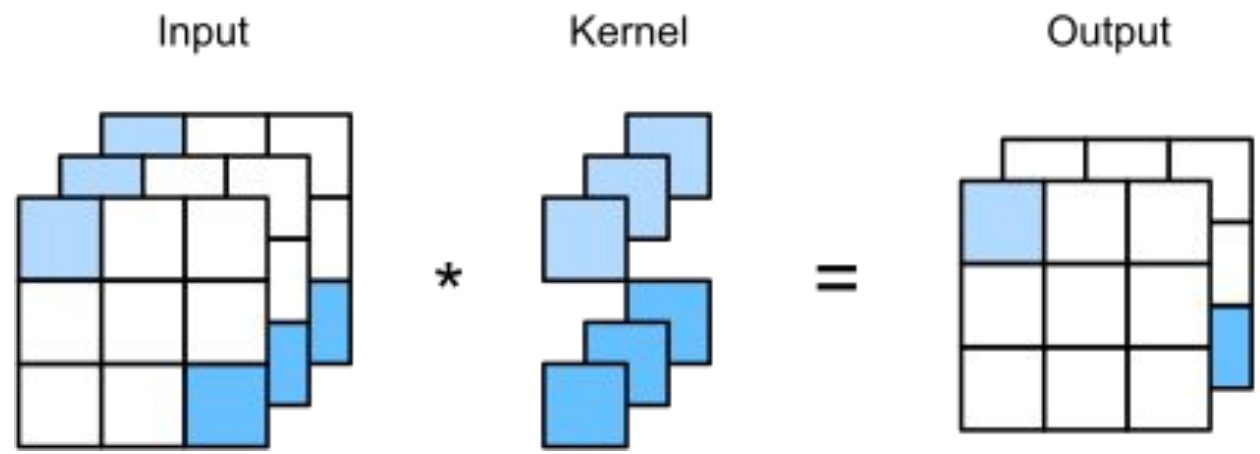
Blue Channel, 2D Matrix

Green Channel, 2D Matrix

Red Channel (n, n) or (n, m) - 2D Matrix

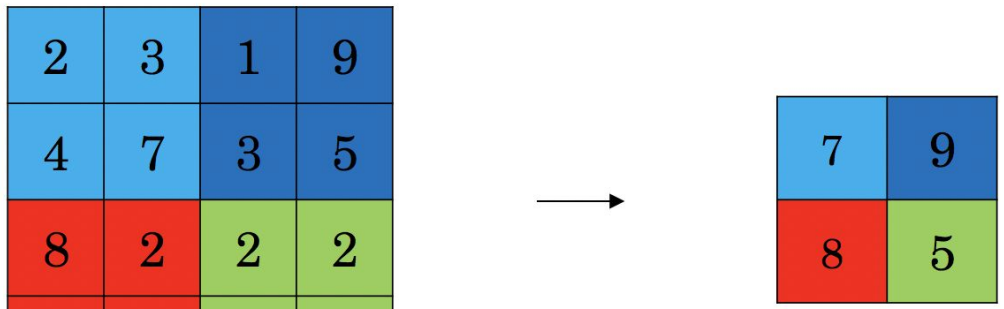
Convolution Operation : Single / Multiple Kernel





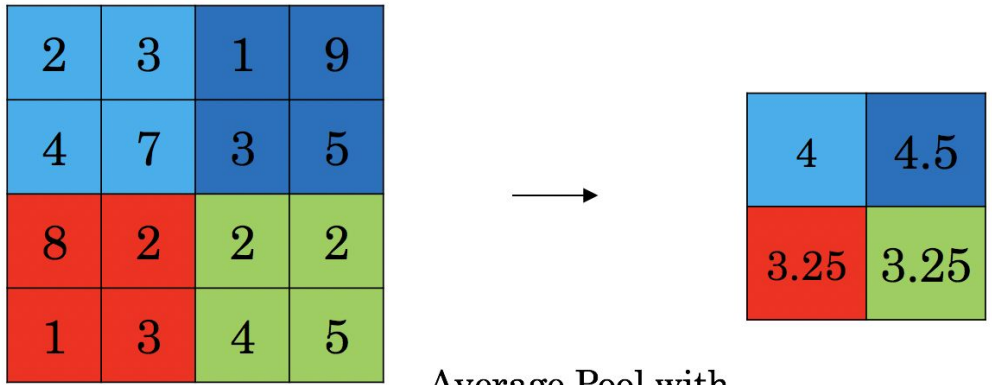
Pooling Operation

Max Pool



Max-Pool with a 2 by 2 filter and stride 2.

Average Pool



Average Pool with a 2 by 2 filter and stride 2.

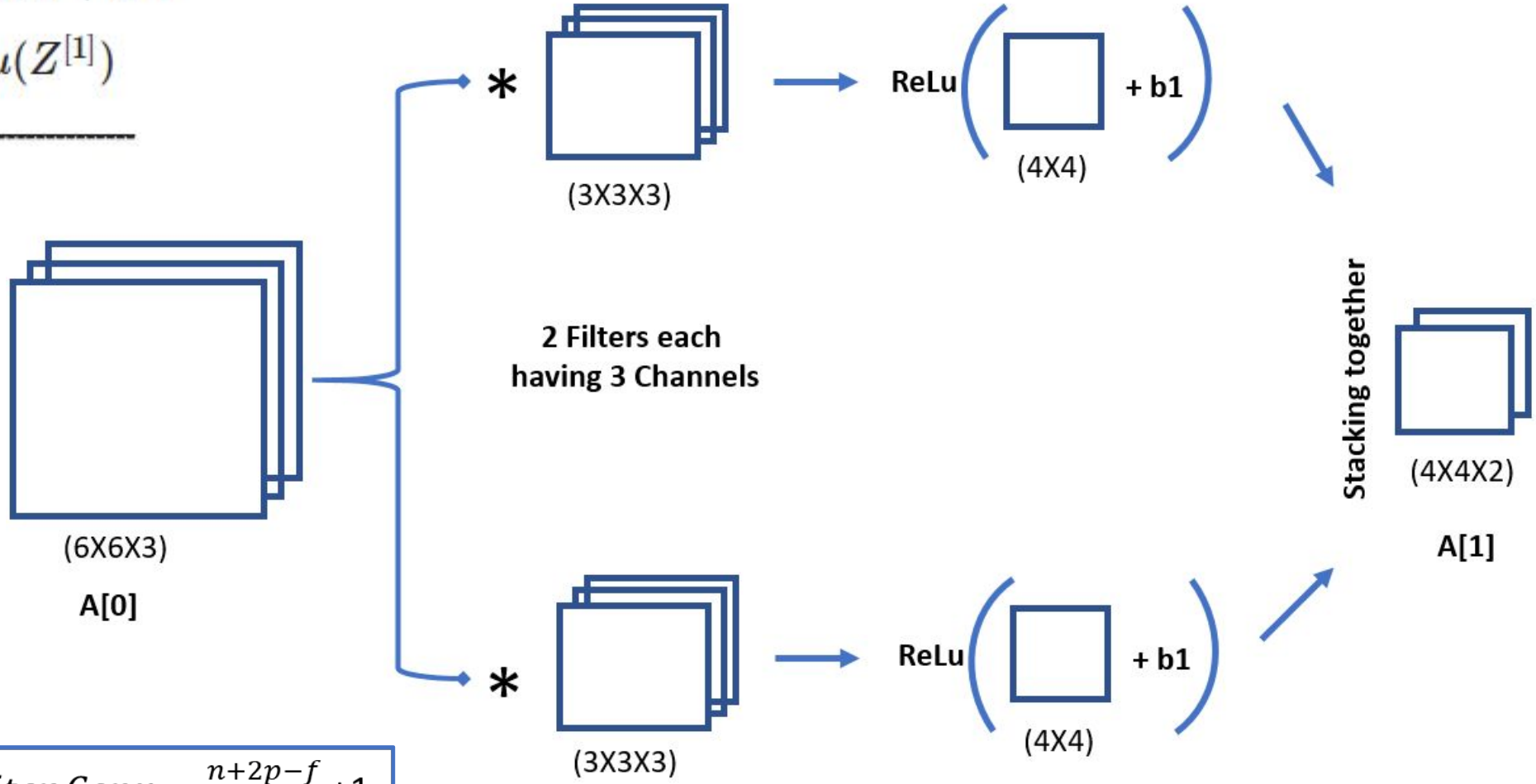
Zero Padding

0	0	0	0	0	0
0	35	19	25	6	0
0	13	22	16	53	0
0	4	3	7	10	0
0	9	8	1	3	0
0	0	0	0	0	0

Deeper look into One Layer of a Convolution

$$Z^{[1]} = W^{[1]} \circledast A^{[0]} + b^{[1]}$$

$$A^{[1]} = ReLu(Z^{[1]})$$



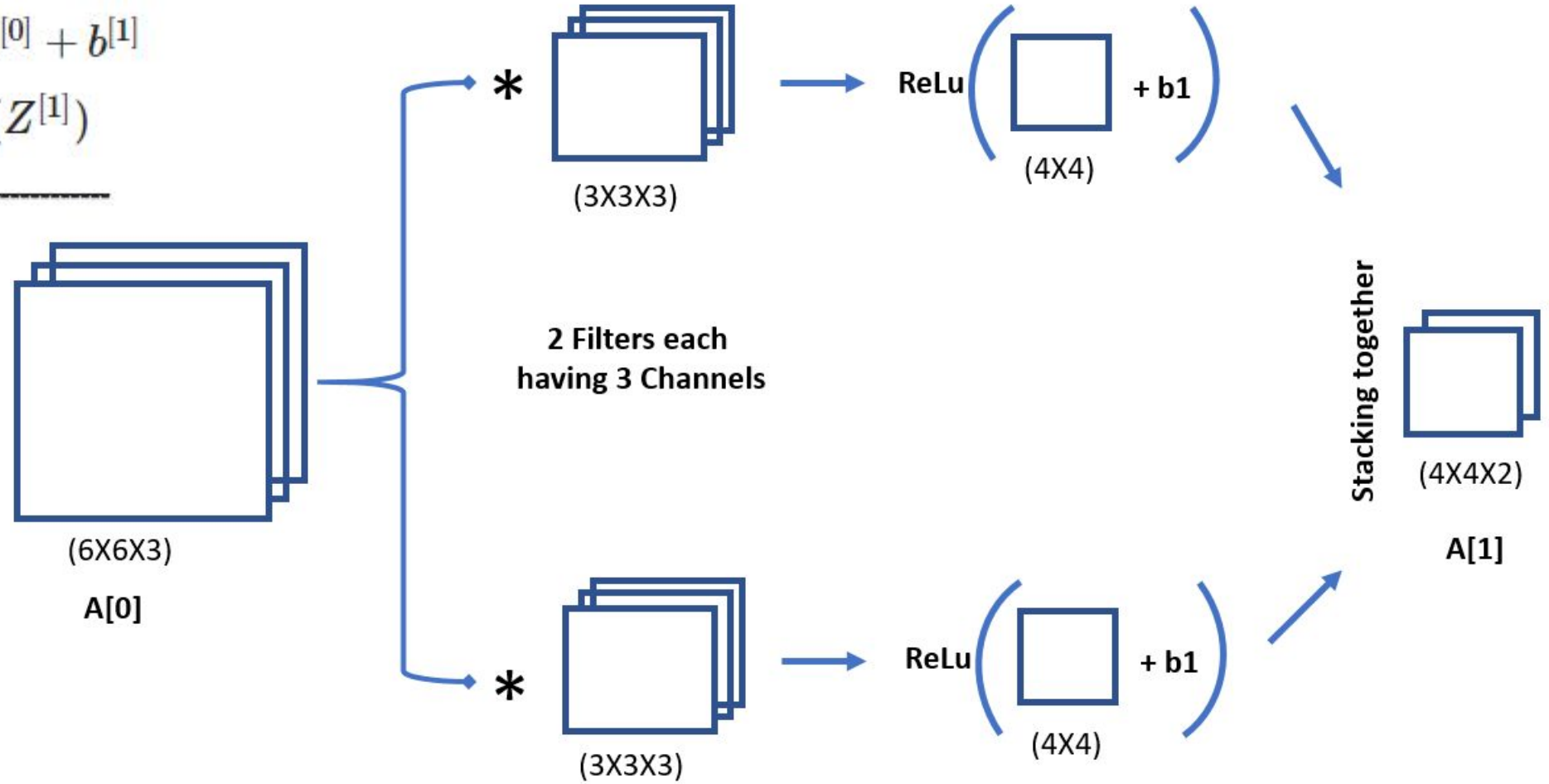
$$Output\ shape\ after\ Conv = \frac{n+2p-f}{s} + 1$$

n= size of input (6), p=padding (0), f=size of filter(3), s=stride(1)

Deeper look into One Layer of a Convolution

$$Z^{[1]} = W^{[1]} \circledast A^{[0]} + b^{[1]}$$

$$A^{[1]} = ReLu(Z^{[1]})$$



$$Output\ shape\ after\ Conv = \frac{n+2p-f}{s} + 1$$

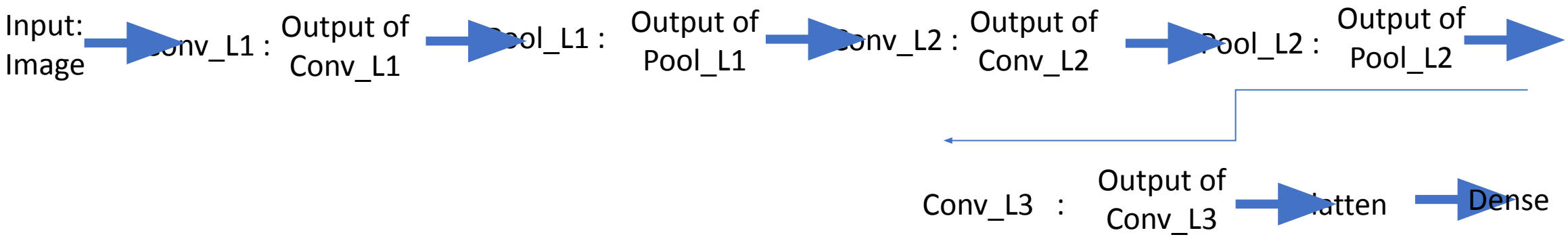
$$\text{Calculating Parameters} = \left(f^l \times f^l \times n_c^{l-1} + 1\right) \times n_f^l$$

f^l = size of filter in layer 'L' : (3), n_c^{l-1} = number of channel in layer 'L-1' : (3), n_f^l = number of filter in layer 'L' : (2)

Building the architecture

LeNet-5: Example of an early ConvNet

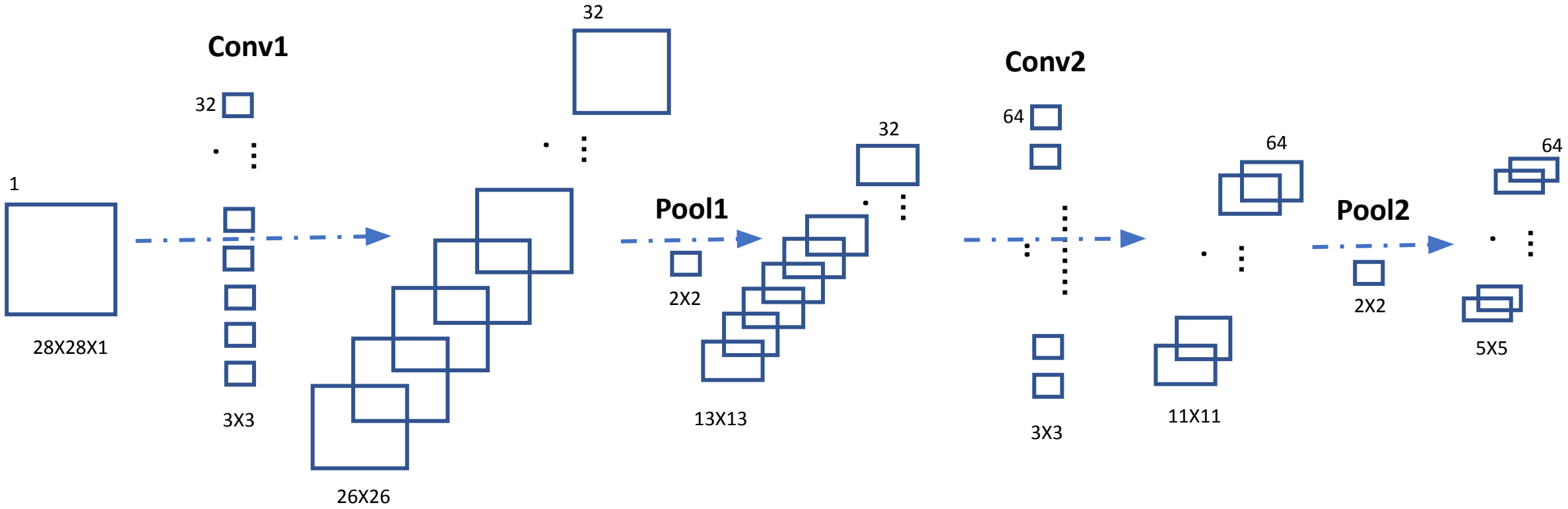
```
# Define convnet
# Q: Which API are we using? A: Functional API
inputs = keras.Input(shape=(28, 28, 1))      # Q: How many channels does the input image have? A: 1
x = layers.Conv2D(filters=32, kernel_size=3, activation="relu")(inputs)  # Q: Meaning of each argument?
x = layers.MaxPooling2D(pool_size=2)(x)      # Q: What is the height and width of feature maps after this layer? A: 13x13
x = layers.Conv2D(filters=64, kernel_size=3, activation="relu")(x)
x = layers.MaxPooling2D(pool_size=2)(x)
x = layers.Conv2D(filters=128, kernel_size=3, activation="relu")(x)
x = layers.Flatten()(x)
outputs = layers.Dense(10, activation="softmax")(x)  # Need Dense layer at the end for classification
model = keras.Model(inputs=inputs, outputs=outputs)
```



CNN Example 1:

P1: $(3 \times 3 \times 1 + 1) \times 32 = 320$

P2: $(3 \times 3 \times 32 + 1) \times 64 = 18496$



Input: Image → Conv_L1 → Output of Conv_L1 → Pool_L1 → Output of Pool_L1 → Conv_L2 → Output of Conv_L2 → Pool_L2 → Output of Pool_L2

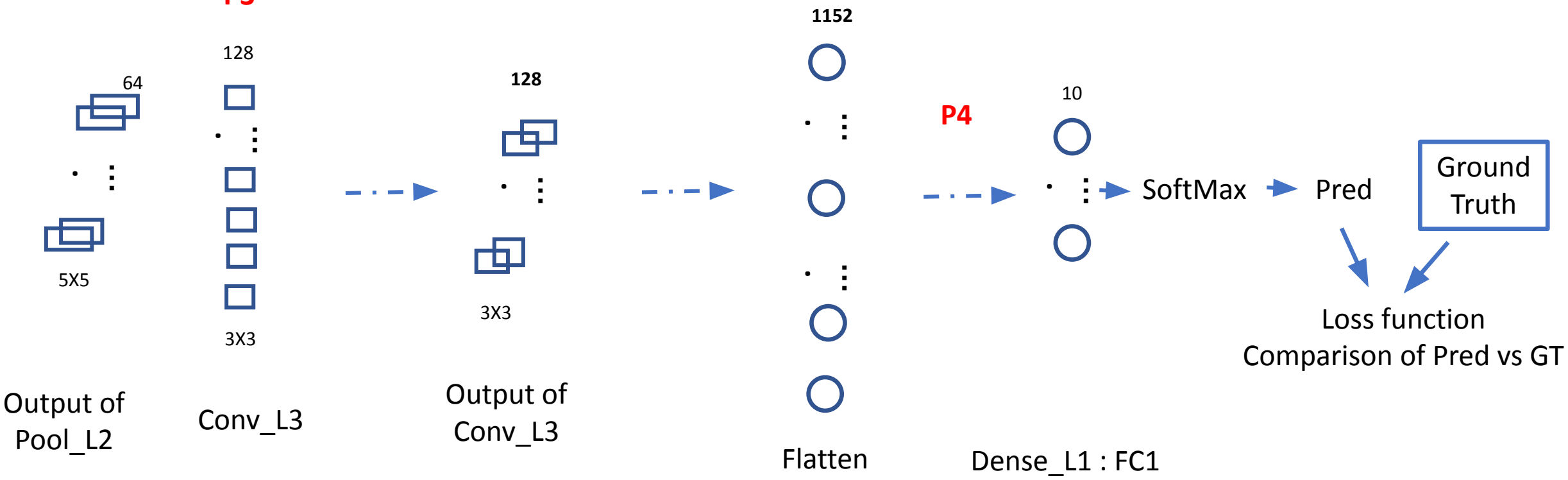
$$Output\ shape\ after\ Conv = \frac{n+2p-f}{s} + 1$$

$$Output\ shape\ after\ Pool = \frac{n-f}{s} + 1$$

Calculating Parameters= $\left(f^l \times f^l \times n_c^{l-1} + 1\right) \times n_f^l$

P3 : (3X3X64+1) X 128 = 73856

P4 : [10, 1152] + b(10) =((10 × 1152) + 10) =11,530



Thanks !