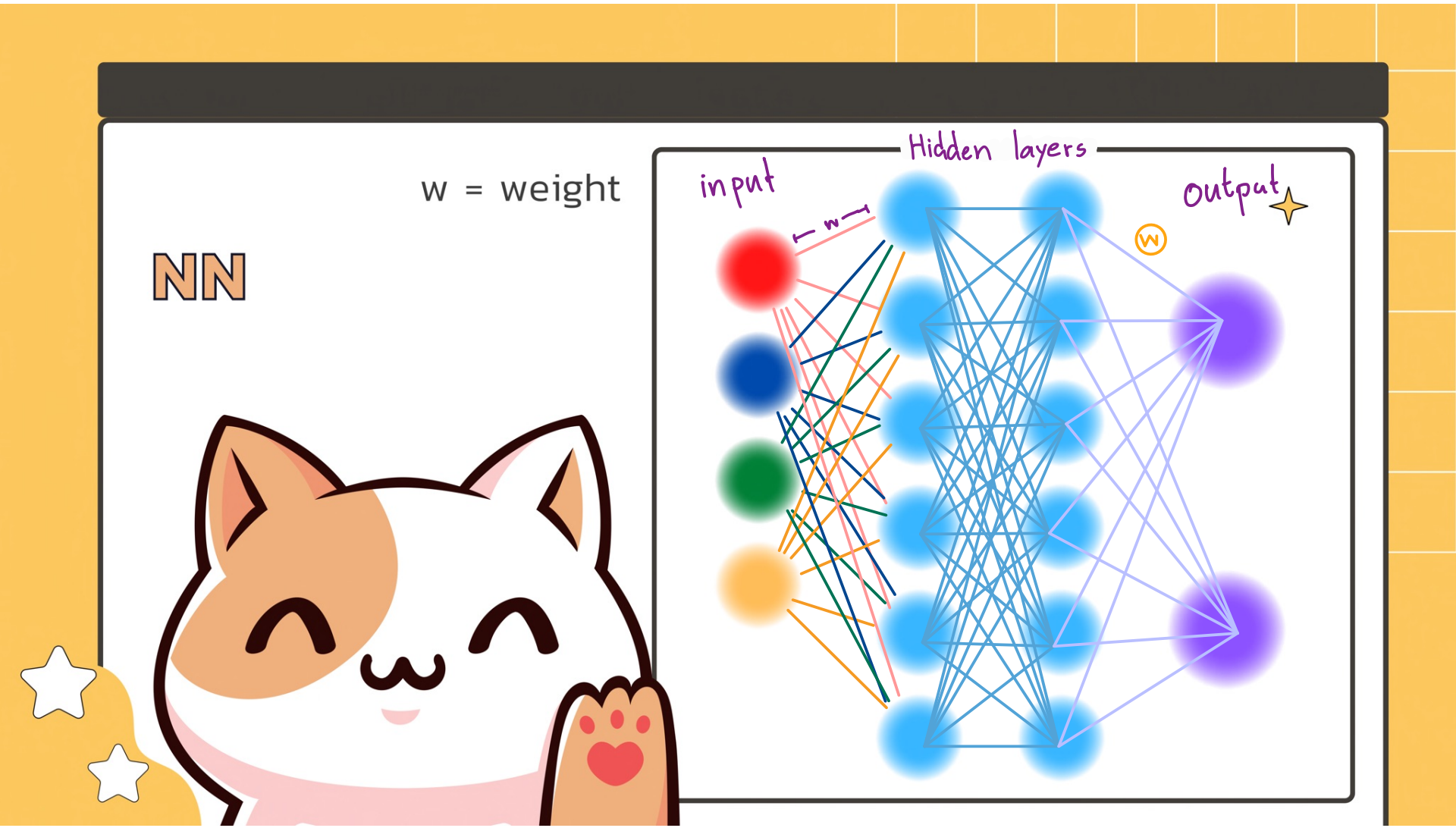
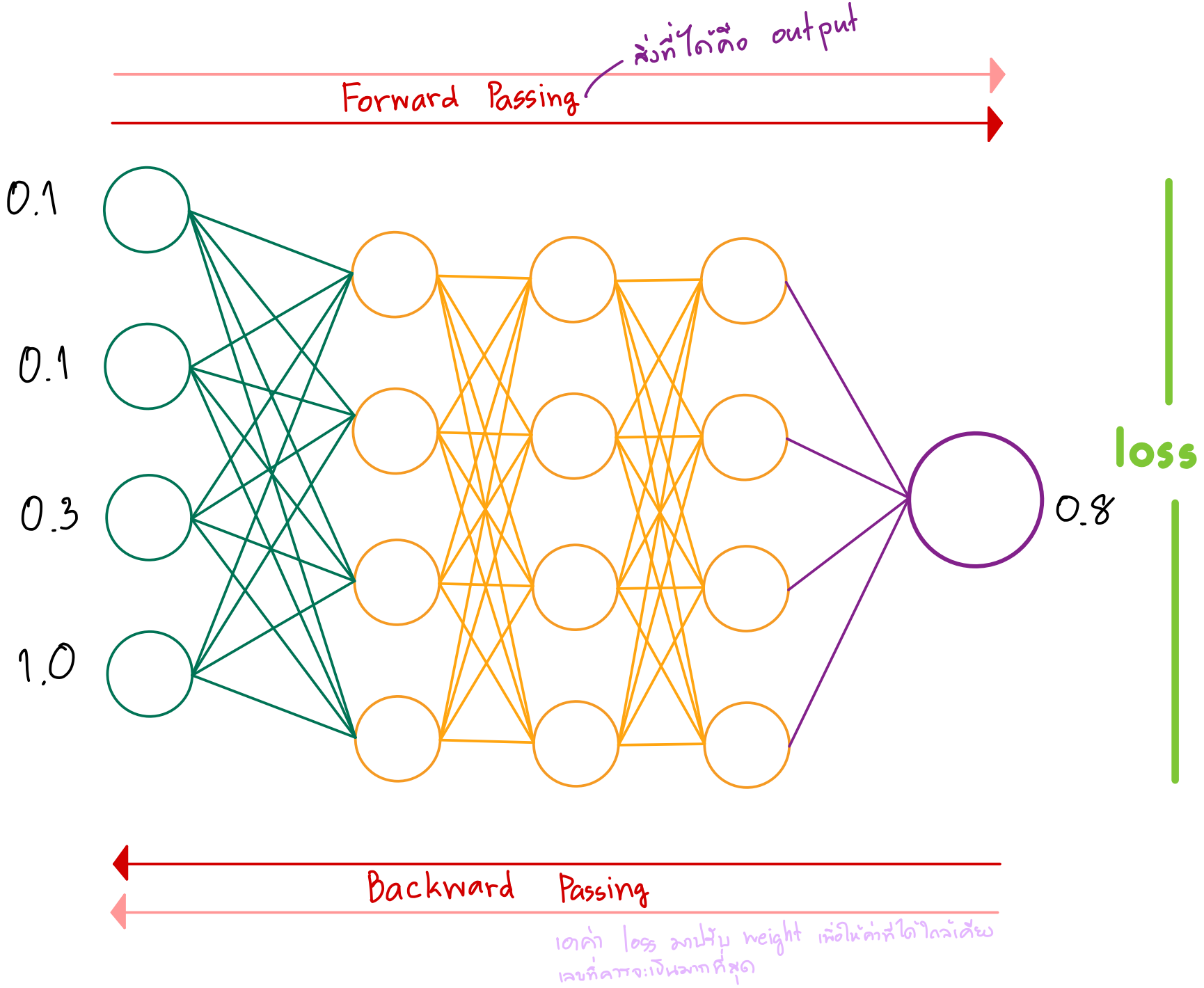


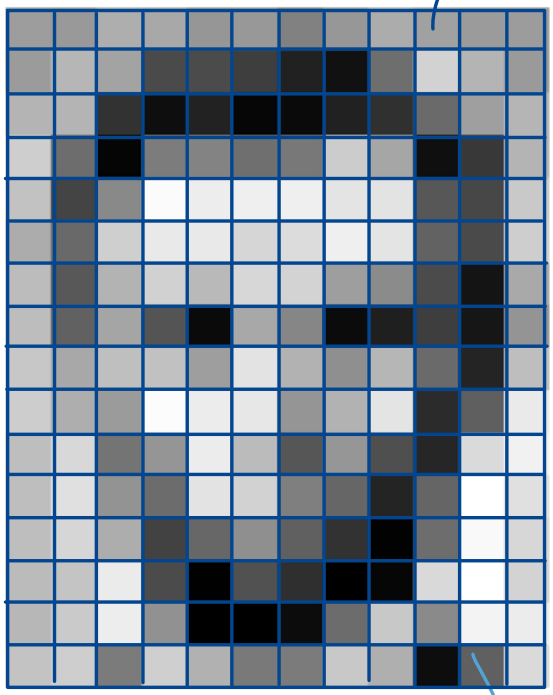
This slide is taken from Andrew Ng

[Krizhevsky et al., 2012]


สิ่งที่ได้คือ output
Forward Passing
0.1
0.1
0.3
1.0
0.8
loss
Backward Passing
เอาค่า loss มาปรับ weight เพื่อให้ค่าที่ได้ใกล้เคียง
เลขที่คาดจะเป็นมากที่สุด

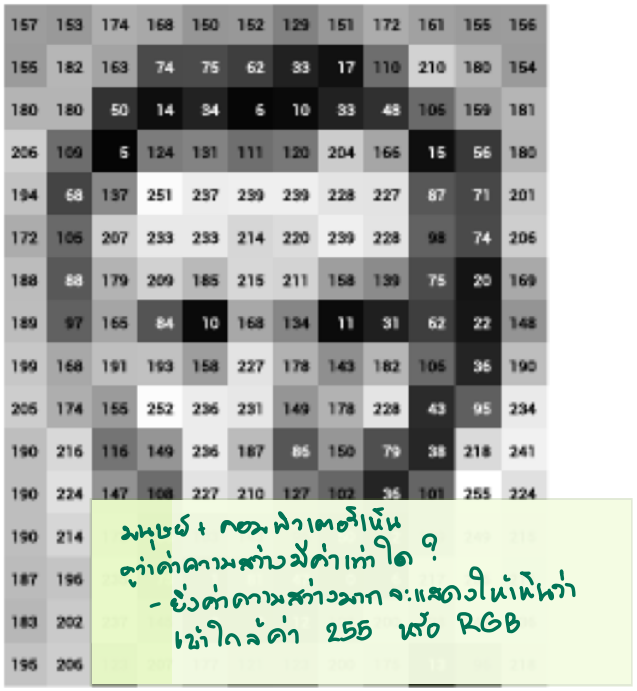
CNNs = Translating pixels to concepts

What you see



Input Image

What you both see



Input Image + values

What the computer "sees"

Pixel intensity values
("pix-el"=picture-element)

An image is just a matrix of numbers [0,255]. i.e., 1080x1080x3 for an RGB image.

Question: is this Lincoln? Washington? Jefferson? Obama?

How can the computer answer this question?

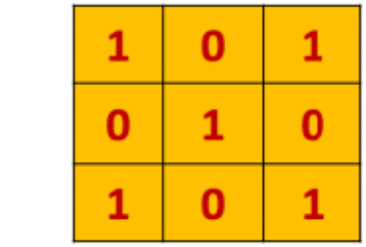
Can I just do classification on the 1,166400-long image vector directly?

No. Instead: exploit image spatial structure. Learn patches. Build them up

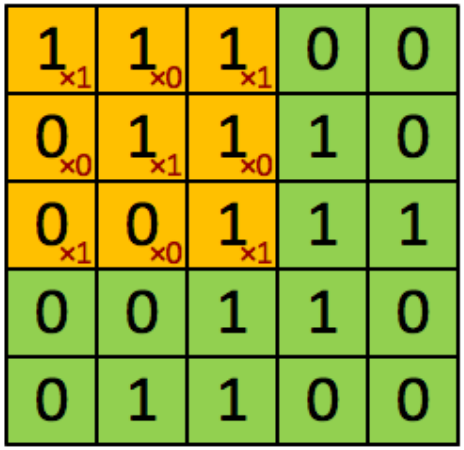
"Representations"

Filters extract Features

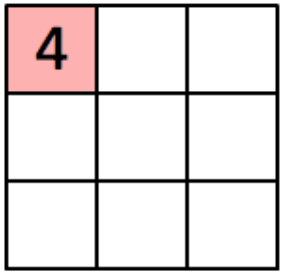
Convolution operation is element wise multiply and add



Filter / Kernel



Image



Convolved Feature

Producing Feature Maps

Original

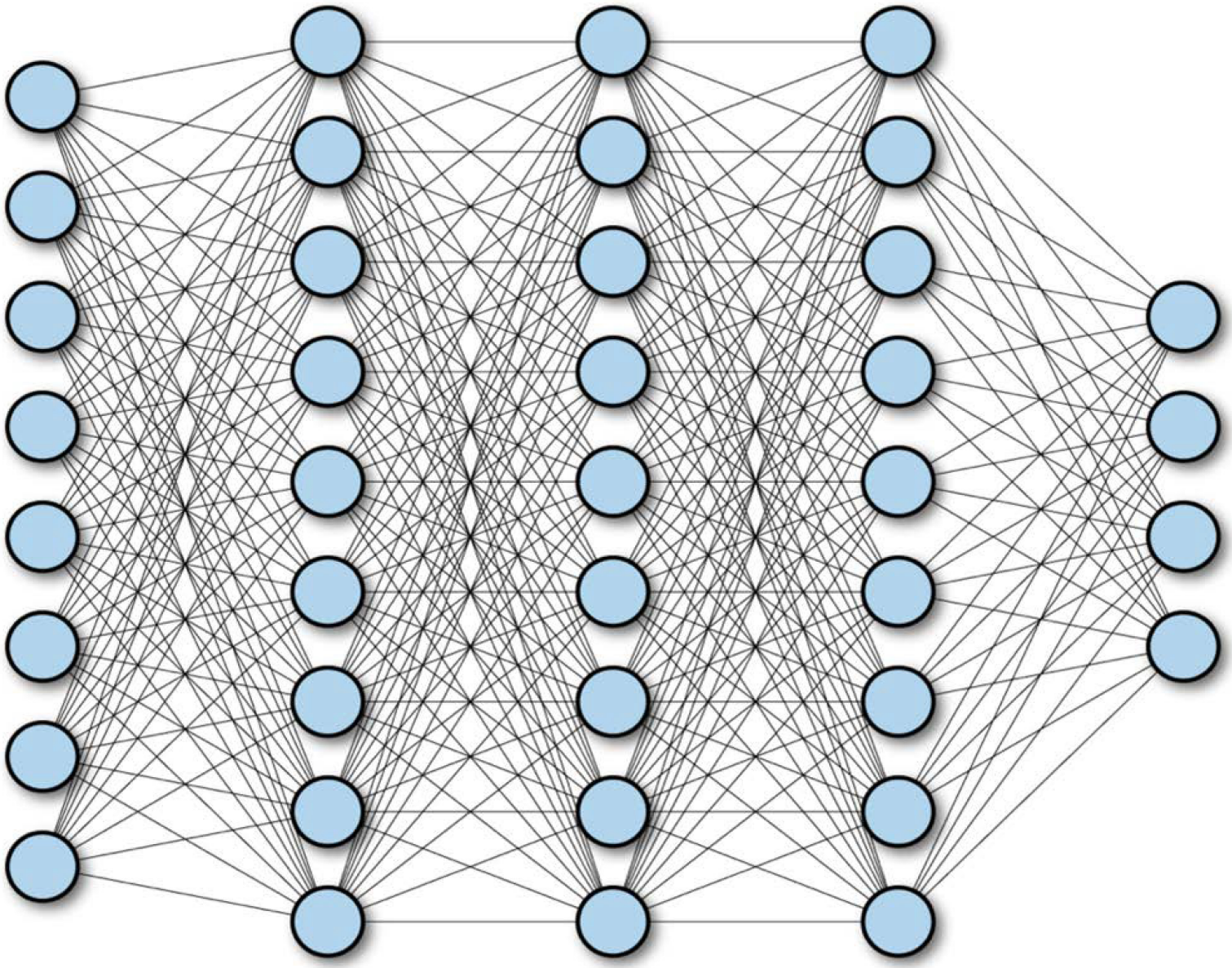
Sharpen

Edge Detect

"Strong" Edge Detect

Filter คือ ตารางที่มันเล็กกว่ารูป
สิ่งที่ Filter ทำคือ จะเอาค่าของสีของ
ไปคูณกับค่าสีที่อยู่ตำแหน่งเดียวกัน

Fully Connected Neural Network



CNNs: Putting all their ingredients together

LeNet-5

- *Gradient Based Learning Applied To Document Recognition - Y. Lecun, L. Bottou, Y. Bengio, P. Haffner; 1998*
- Helped establish how we use CNNs today
- Replaced manual feature extraction

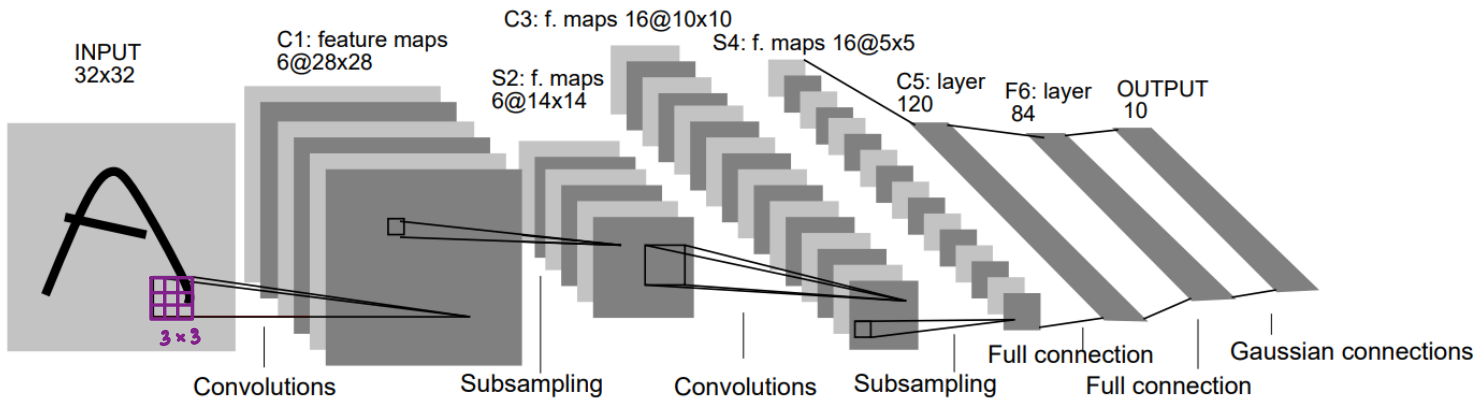
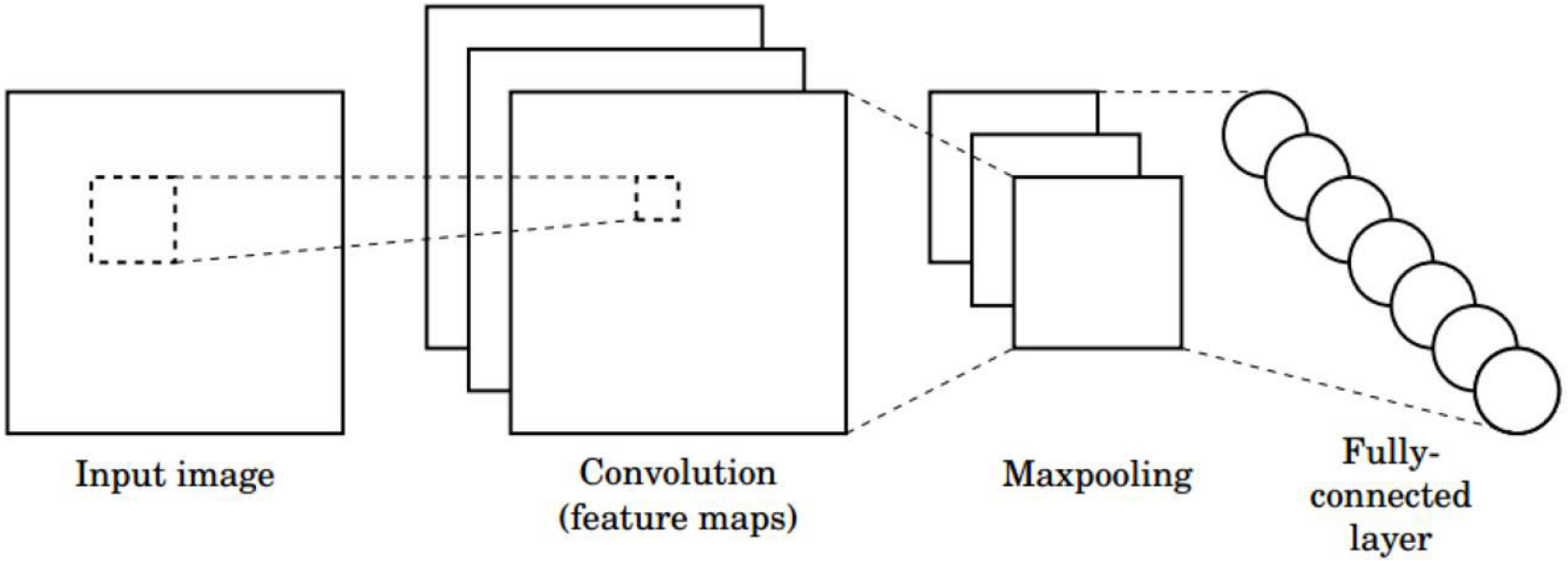


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

[LeCun et al., 1998]

CNNs for Classification



1. **Convolution**: Apply filters to generate feature maps.
2. **Non-linearity**: Often ReLU.
3. **Pooling**: Downsampling operation on each feature map.

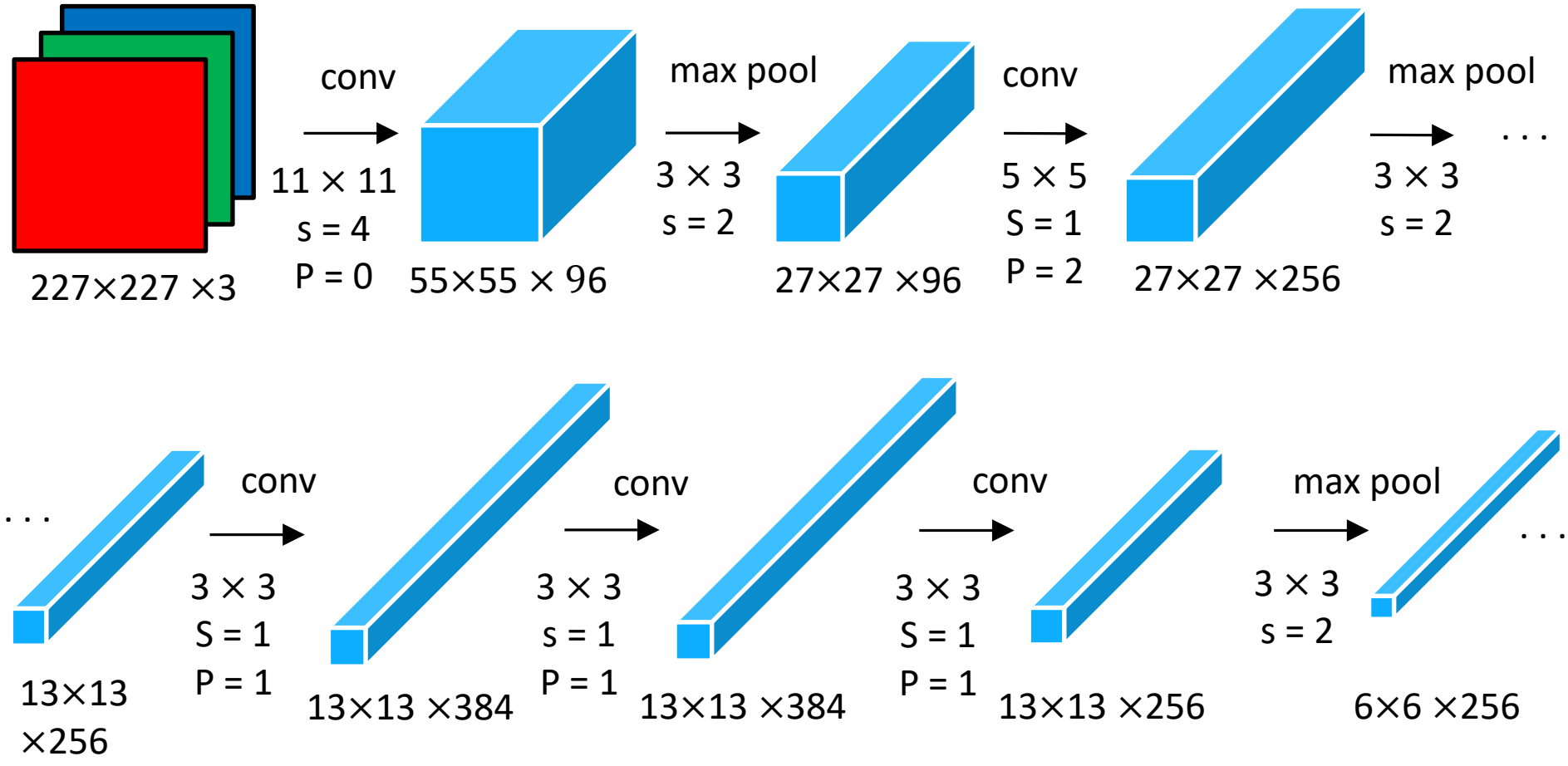
Train model with image data.
Learn weights of filters in convolutional layers.

`tf.keras.layers.Conv2D`

`tf.keras.activations.*`

`tf.keras.layers.MaxPool2D`

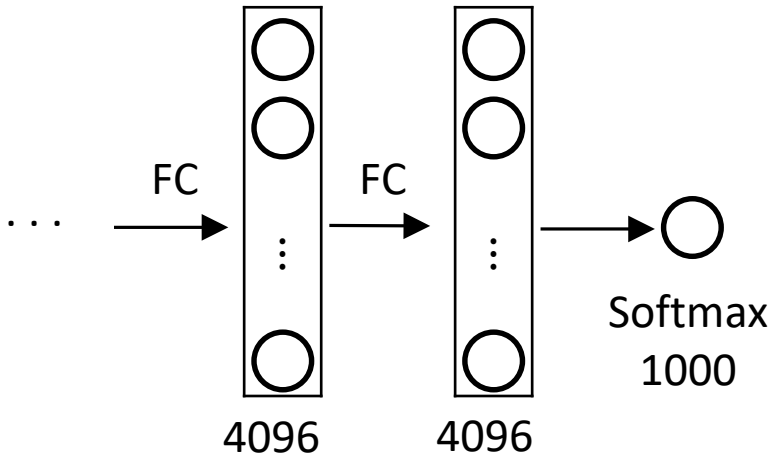
AlexNet



This slide is taken from Andrew Ng

[Krizhevsky et al., 2012]

AlexNet



This slide is taken from Andrew Ng

[**Krizhevsky et al.**, **2012**]

```
from threading import active_count
model = keras.Sequential(
    [
        keras.Input(shape=(4,)),
        layers.Dense(2, activation="relu", name="layer1"),
        layers.Dense(3, activation="relu", name="layer2"),
        layers.Dense(4, name="output")
    ]
)
```

```
model.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 layer1 (Dense)              (None, 2)                 10

 layer2 (Dense)              (None, 3)                 9

 output (Dense)              (None, 4)                 16

=================================================================
Total params: 35
Trainable params: 35
Non-trainable params: 0
_________________________________________________________________
```

Bias = 1