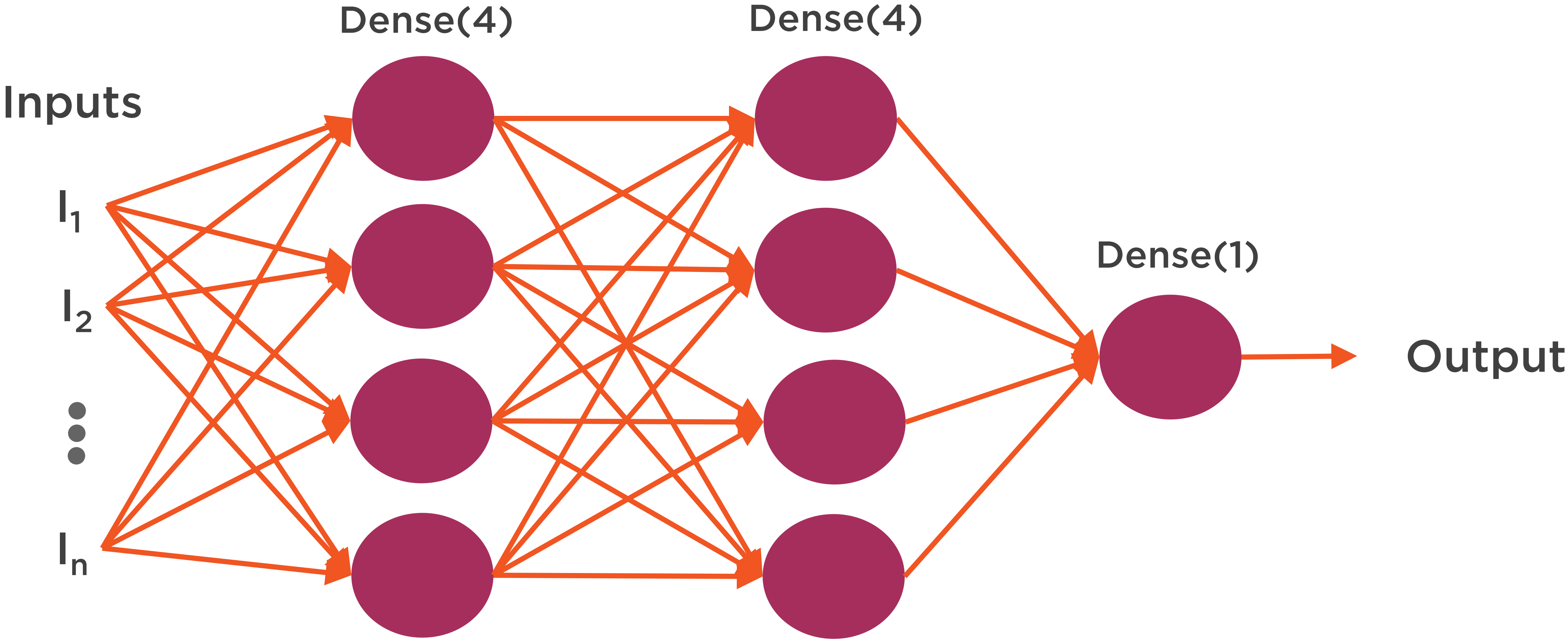
Building Convolutional NN with Keras

Jerry Kurata
CONSULTANT

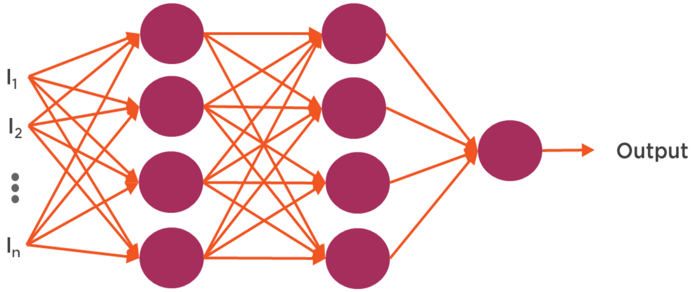
@jerrykur

… a **convolutional neural network** (**CNN**, or **ConvNet**) is a class of deep, feed-forward artificial NN that has successfully been applied to analyzing visual imagery.

Wikipedia


Inputs
I1
I2
In
Dense(4)
Dense(4)
Dense(1)
Output

Curse of Dimensionality



More connections

More weights to train

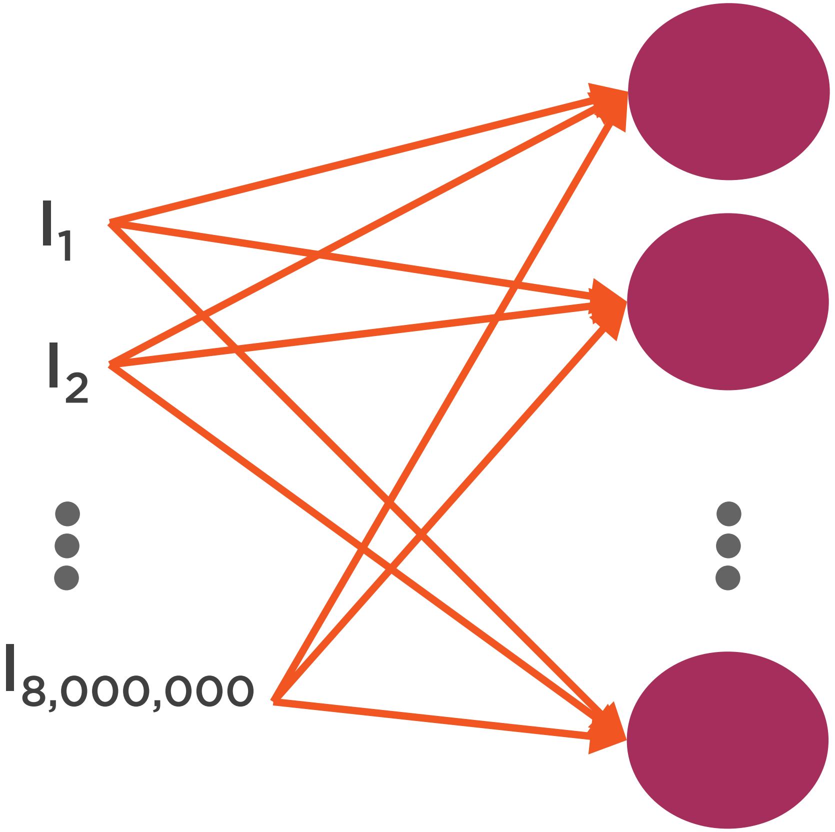
Longer training

Curse of Dimensionality

Inputs

(8 MP)

Dense(1000)



8,000,000 pixels
X 1,000 neurons

8,000,000,000 weights
X 3 colors

24,000,000,000 weights

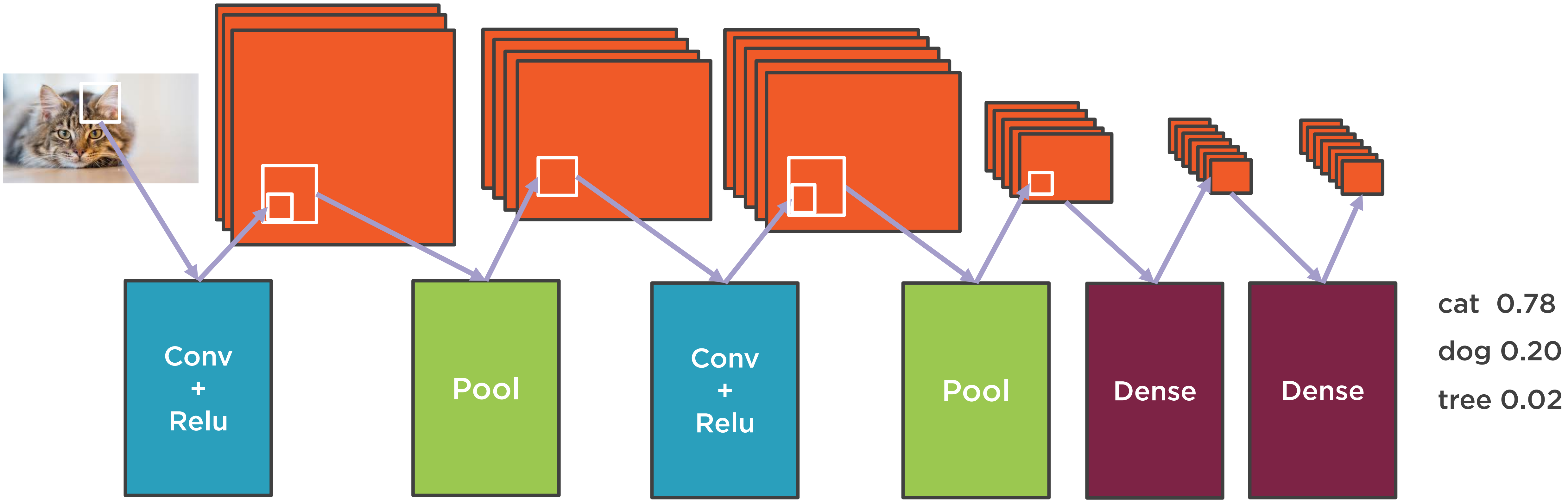
Translational Invariance

Convolutional Neural Network

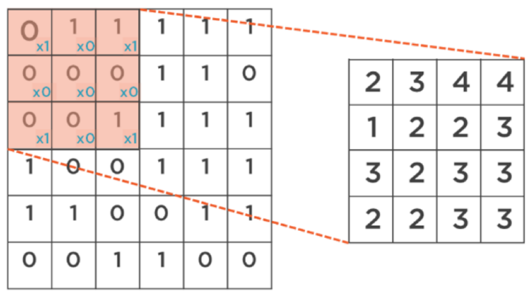
Reduce weights to train

General object identification

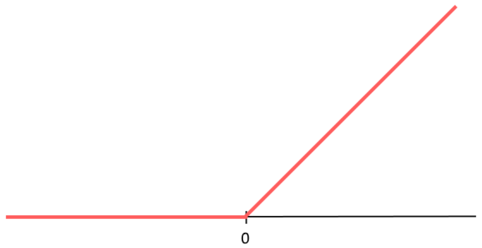
Convolutional Neural Network



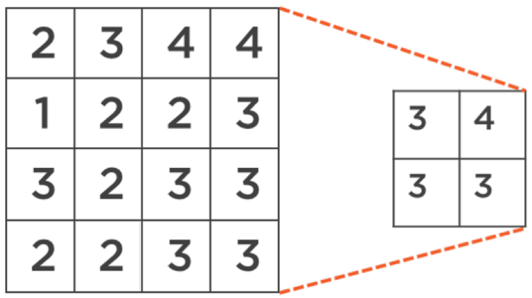
CNN Components



Convolution



Non-linearity (ReLU)



Pooling

Cat 0.6
Dog 0.25
Tree 0.15

Classification

Convolution

Extracts features from image

Preserves feature spatial relationships

- Edges
- Composite elements (nose, eye)

Reduced computation

Filters

Values not fixed

Values are what we train

Improved through training

Trained on labeled images

Detect unique features that determine objects

CNN training faster since only filter weights trained

Weights shared across image

Key Convolution Hyperparameters

Kernel size

Number of filters

Stride

Padding

Kernel Size and Number of Filters

Kernel size -> related pixels	Number of Filters -> Feature Detected
3X3, 5X5, 7X7	16, 32, 64, 96

kernel_size = (2,2)
filters = 5

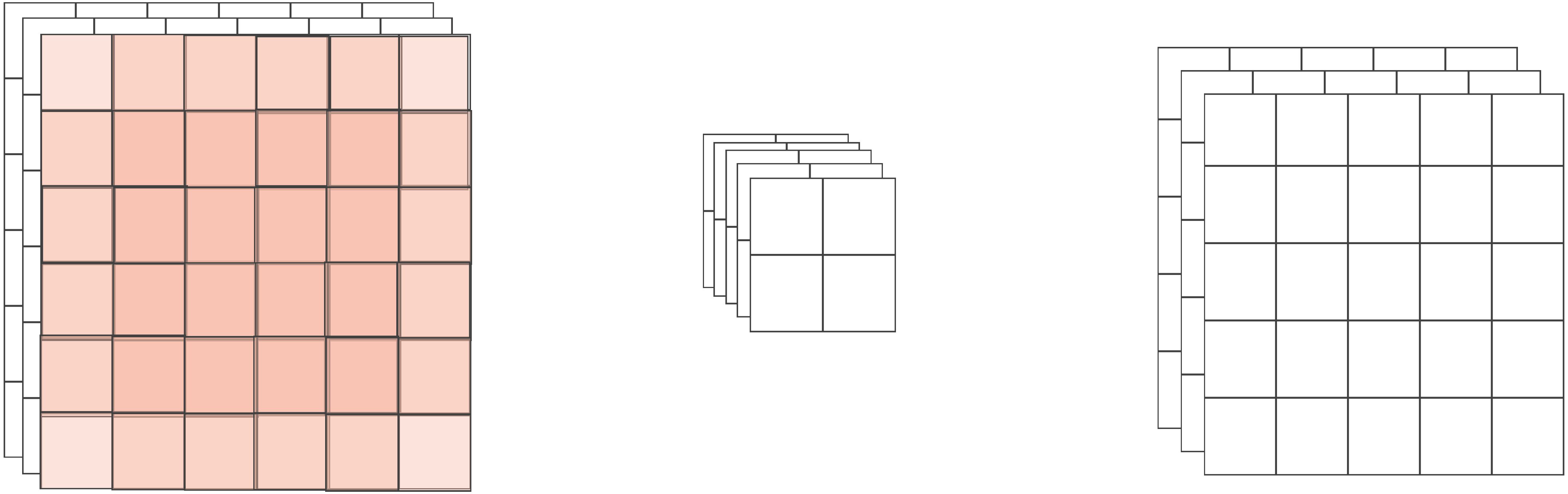


Image	Kernel (Filters)	Feature Maps
6 X 6 X 3	2 X 2 X 5	5 X 5 X 5

Stride

Distance to move filter

Larger stride -> pixel independence

Stride

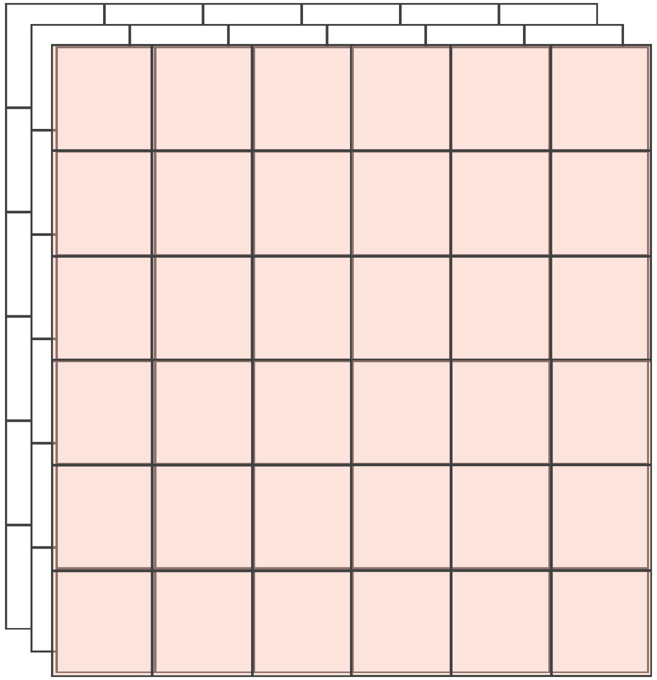
Distance to move filter

Larger values faster

- Decrease size of feature map
- Reduces information passed to next layer

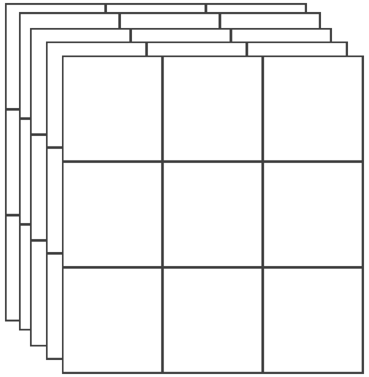
1 is common value

Stride = 3



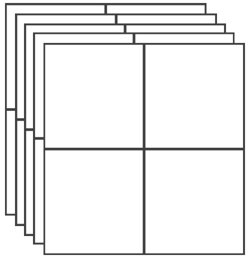
Image

6 X 6 X 3



Kernel (Filters)

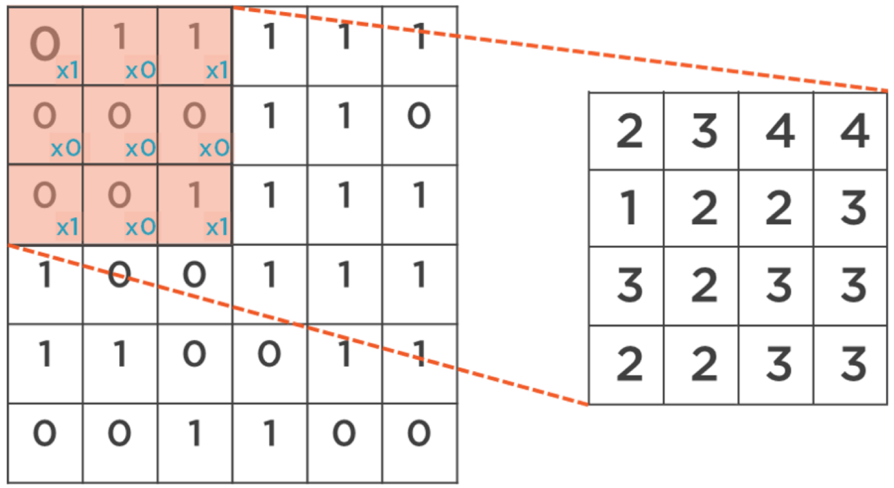
3 X 3 X 5



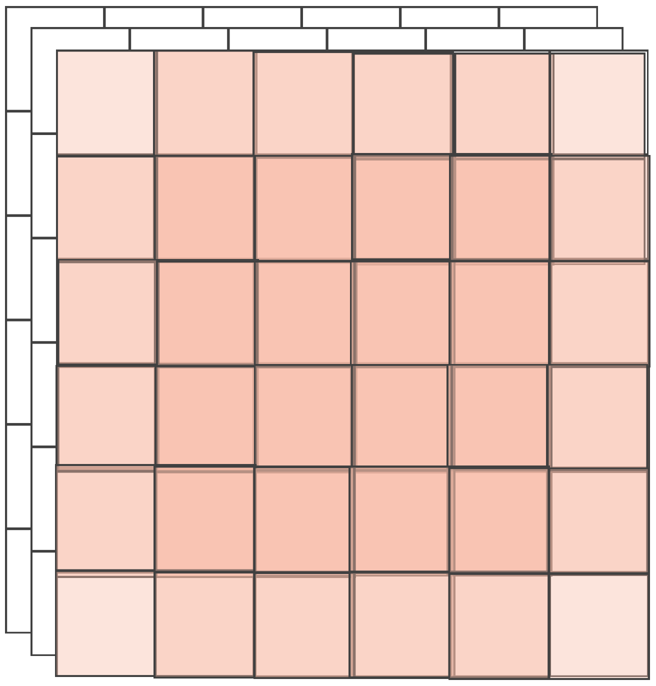
Feature Maps

2 X 2 X 5

Issues with Convolution

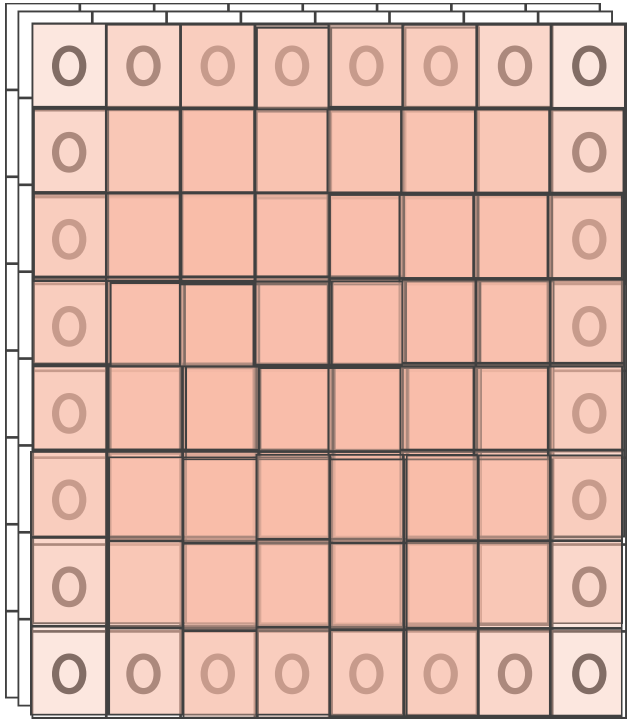


Reduction in spatial dimensions



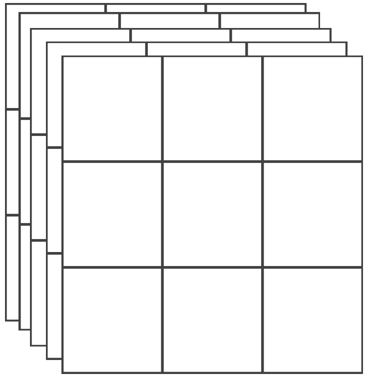
Data at edges is used less

Padding = ‘same’



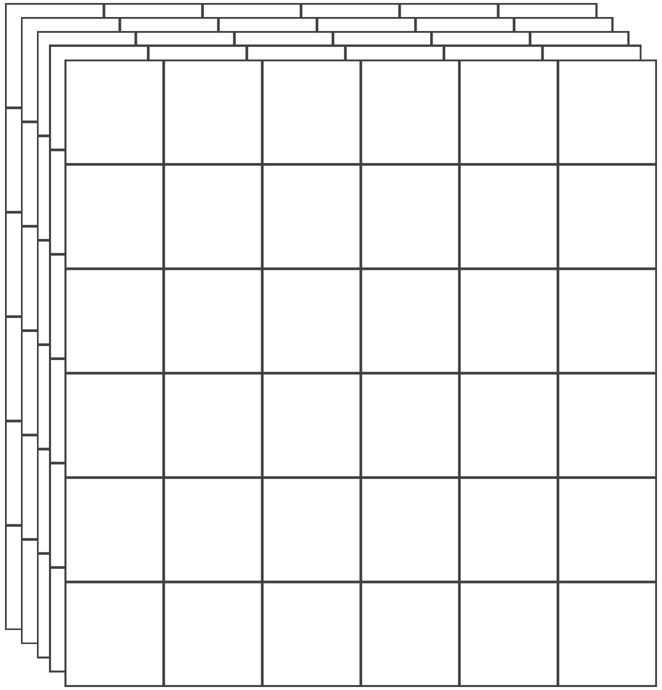
Image

6 X 6 X 3
+ padding



Kernel (Filters)

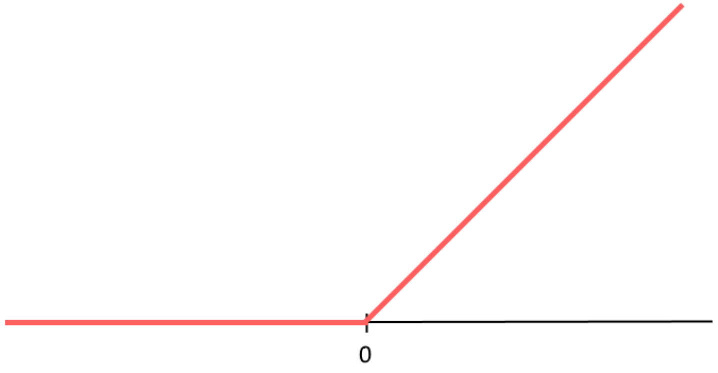
3 X 3 X 5



Feature Maps

6 X 6 X 5

Non-Linear Activation Function



Lets NN handle non-linear

Added in two ways

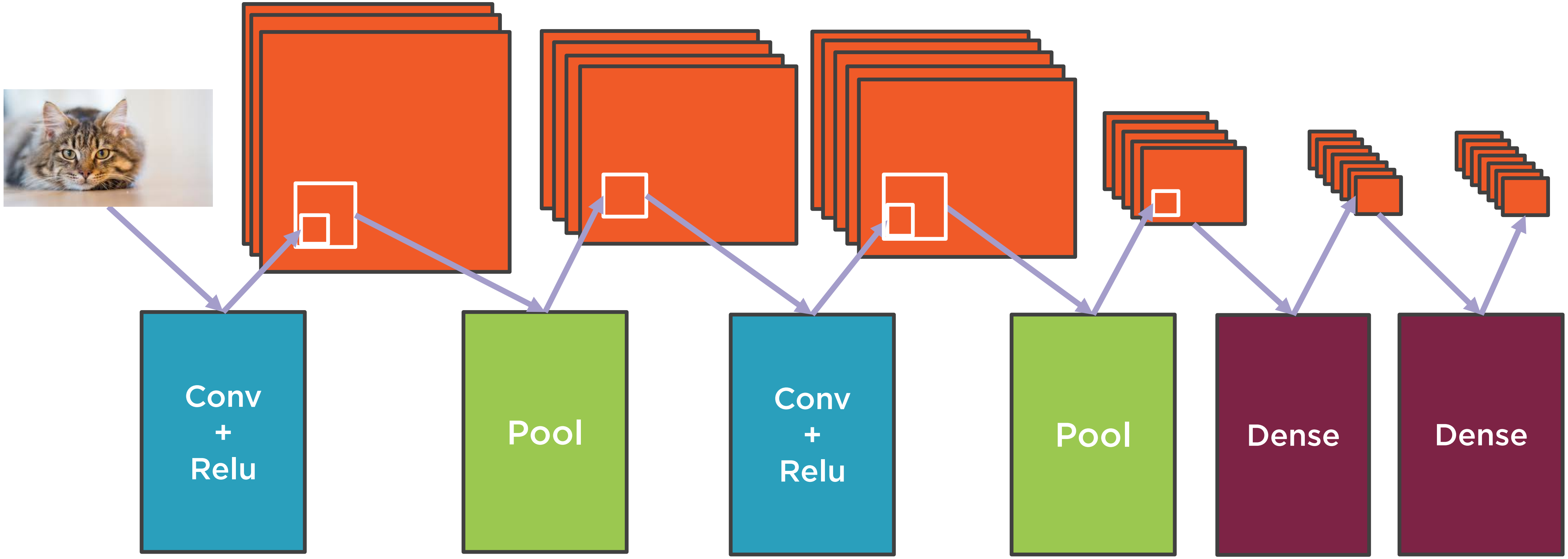
- As a layer after convolution layer
- As parameter to convolution layer

ReLU most common non-linearity function

- $y = \max(0, x)$
- If $(x < 0)$ return 0 else return x

Prevents vanishing gradient

Convolutional Neural Network



Pooling Layer

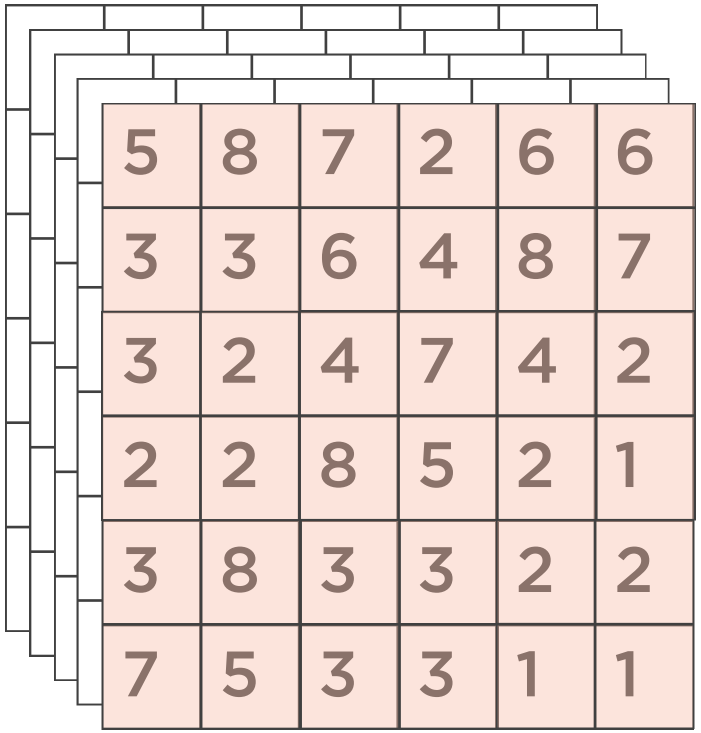
3	5	7	2	3	3
8	1	6	1	1	4
5	6	9	3	2	9
3	2	9	5	7	9
5	4	7	4	8	9
2	5	8	3	3	1

8	7	4
6	9	

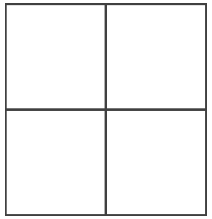
Reduce Dimensionality

Translational Invariance

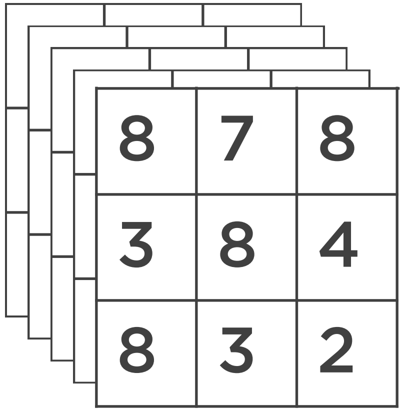
Max Pooling



6 X 6 X 5



2 X 2



$W_{out} = ((W_{in} - F)/S) + 1$

$H_{out} = ((H_{in} - F)/S) + 1$

$D_{in} = D_{out}$

Max Pooling

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

6 X 6 X 5

2 X 2

8	7	8
3	8	4
8	3	2

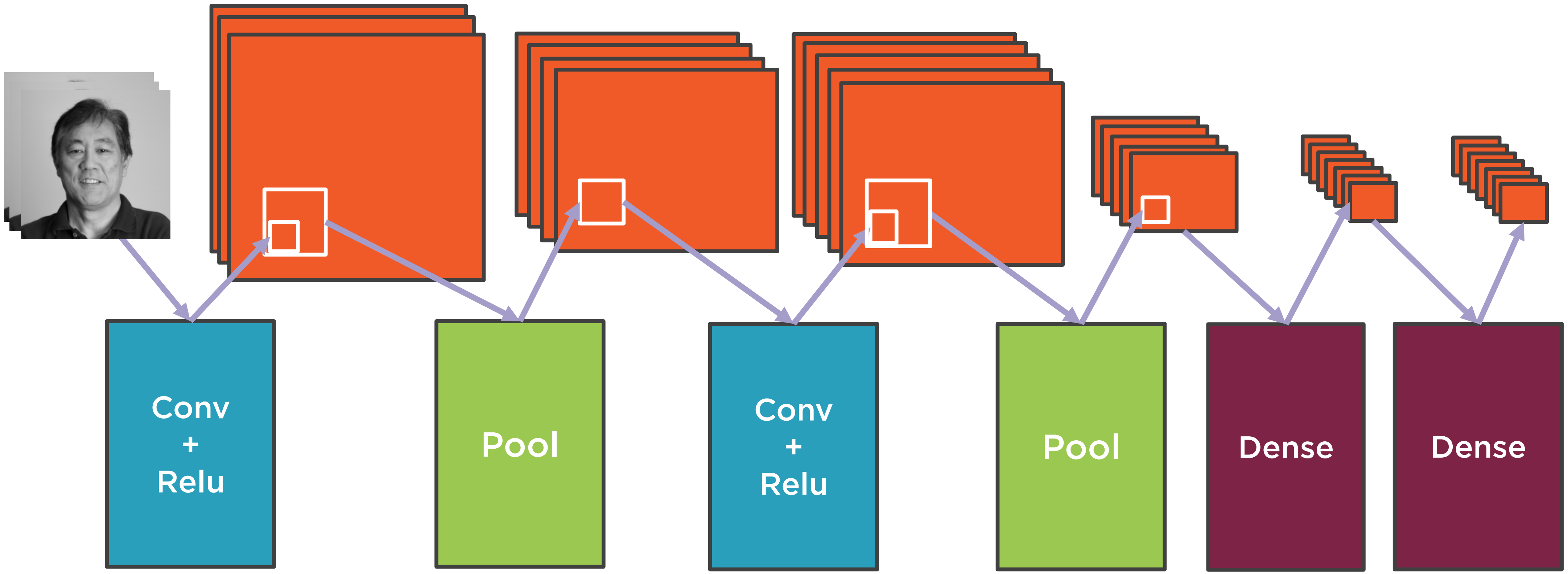
3 X 3 X 5

$W_{out} = ((6 - 2)/2) + 1 = 3$

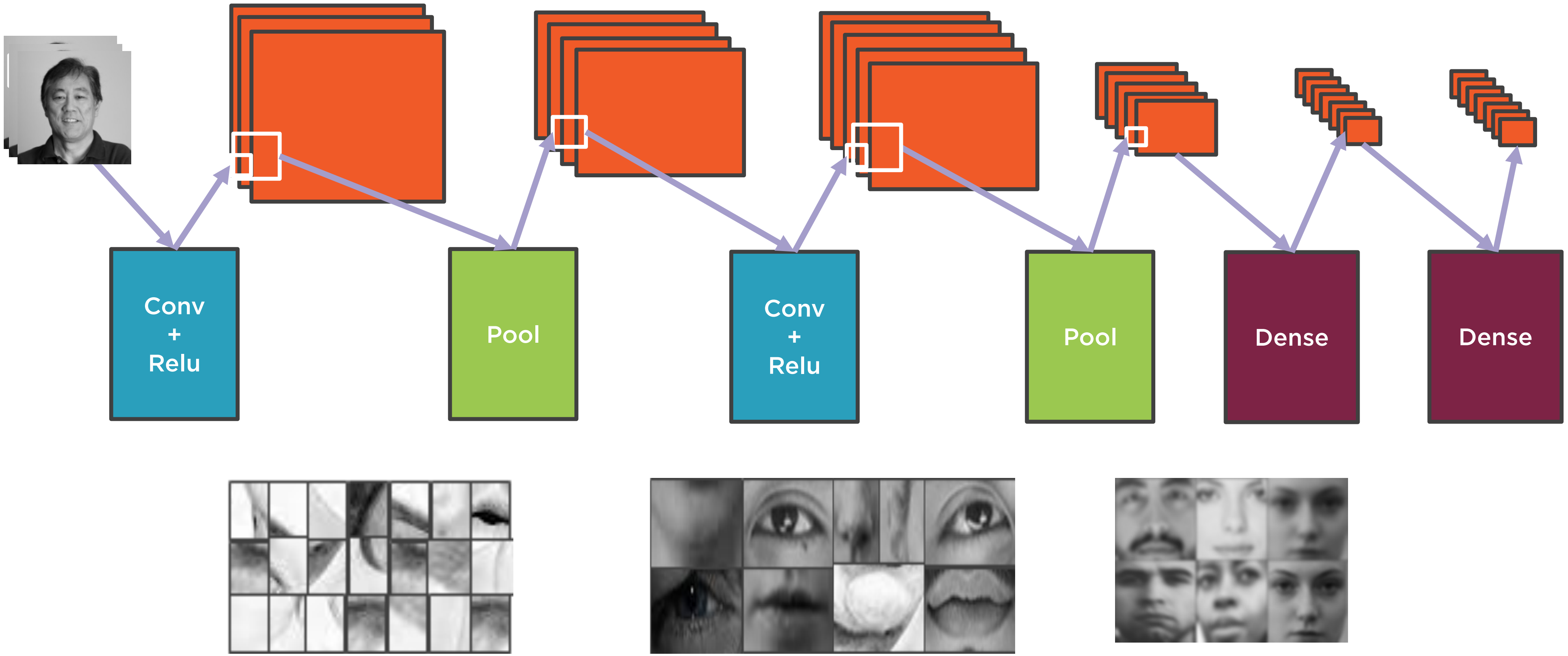
$H_{out} = ((6 - 2)/2) + 1 = 3$

$D_{in} = D_{out} = 5$

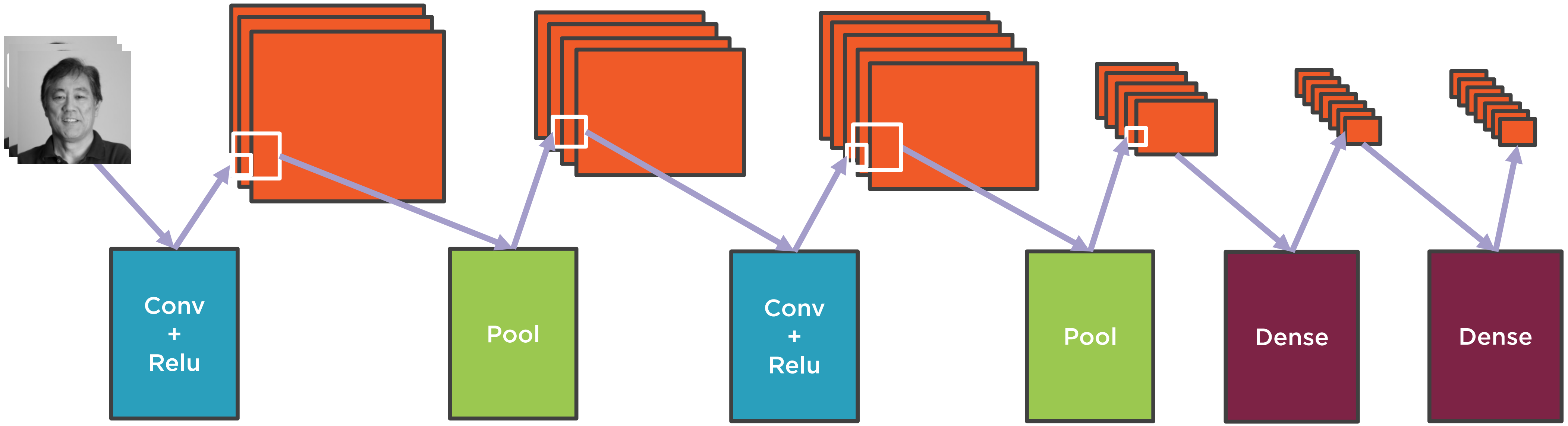
Convolutional Neural Network



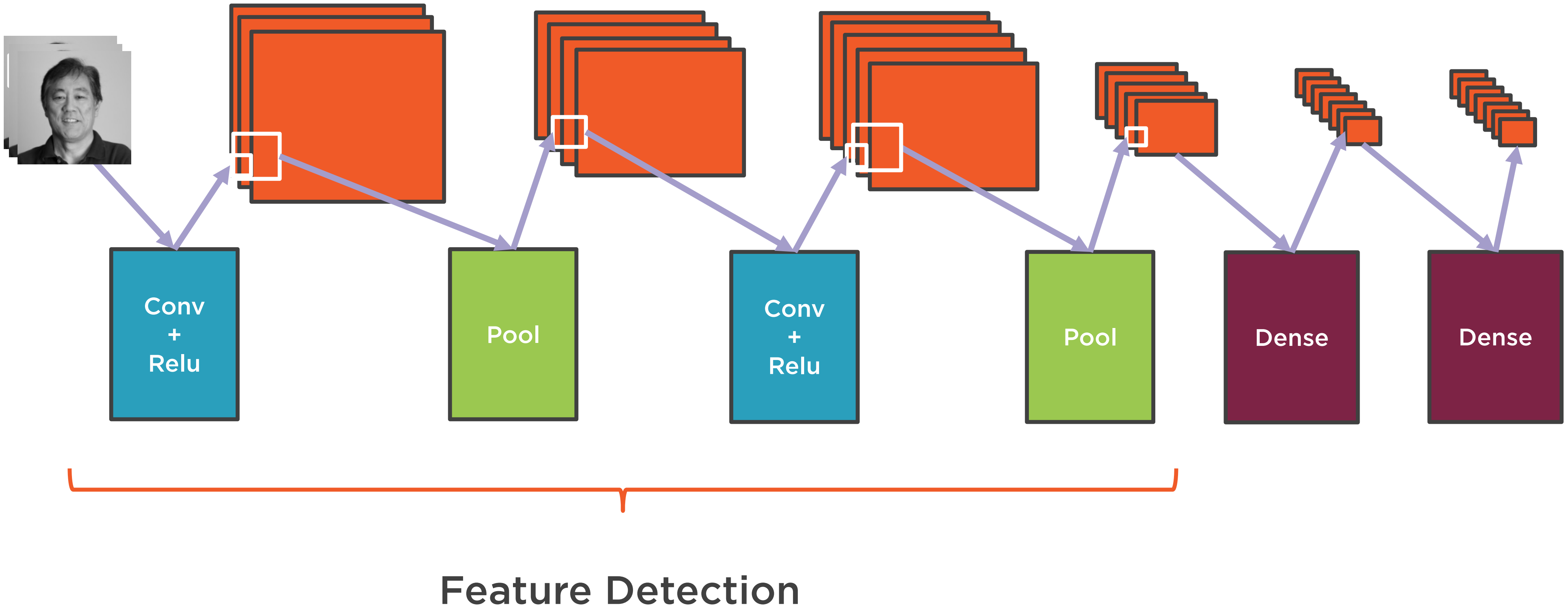
Convolutional Neural Network



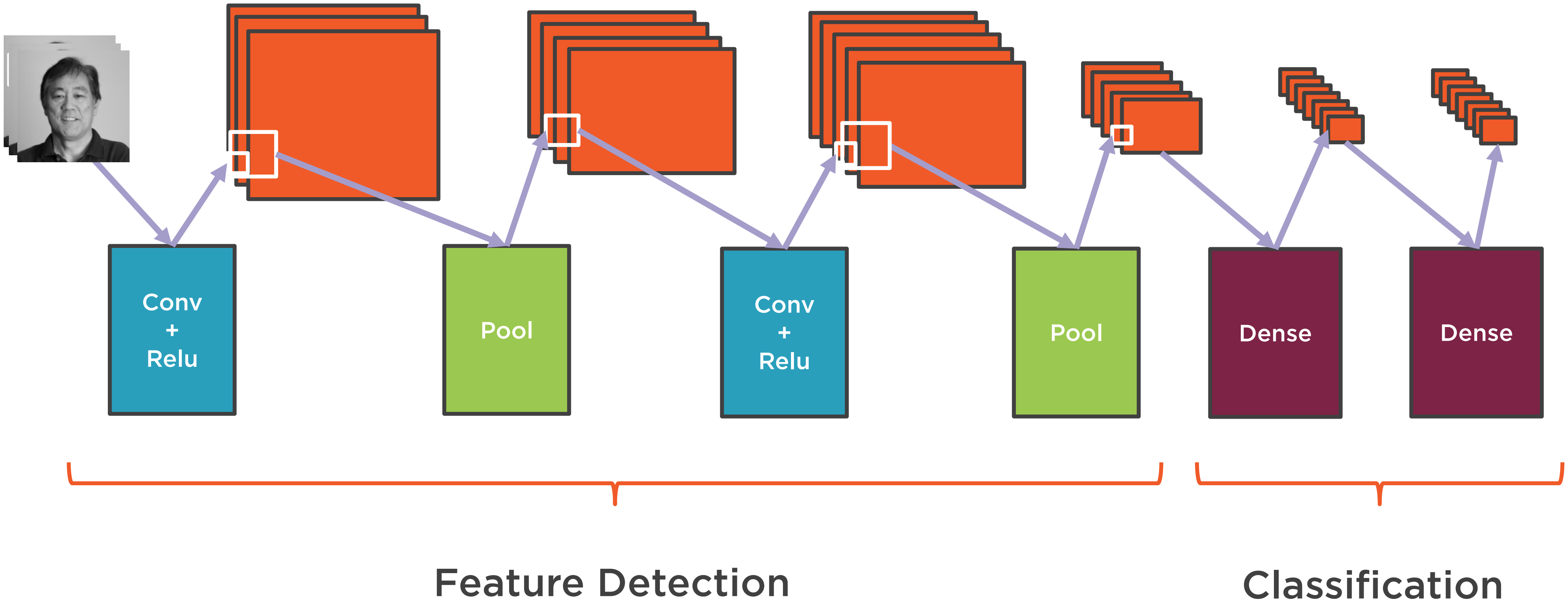
Convolutional Neural Network



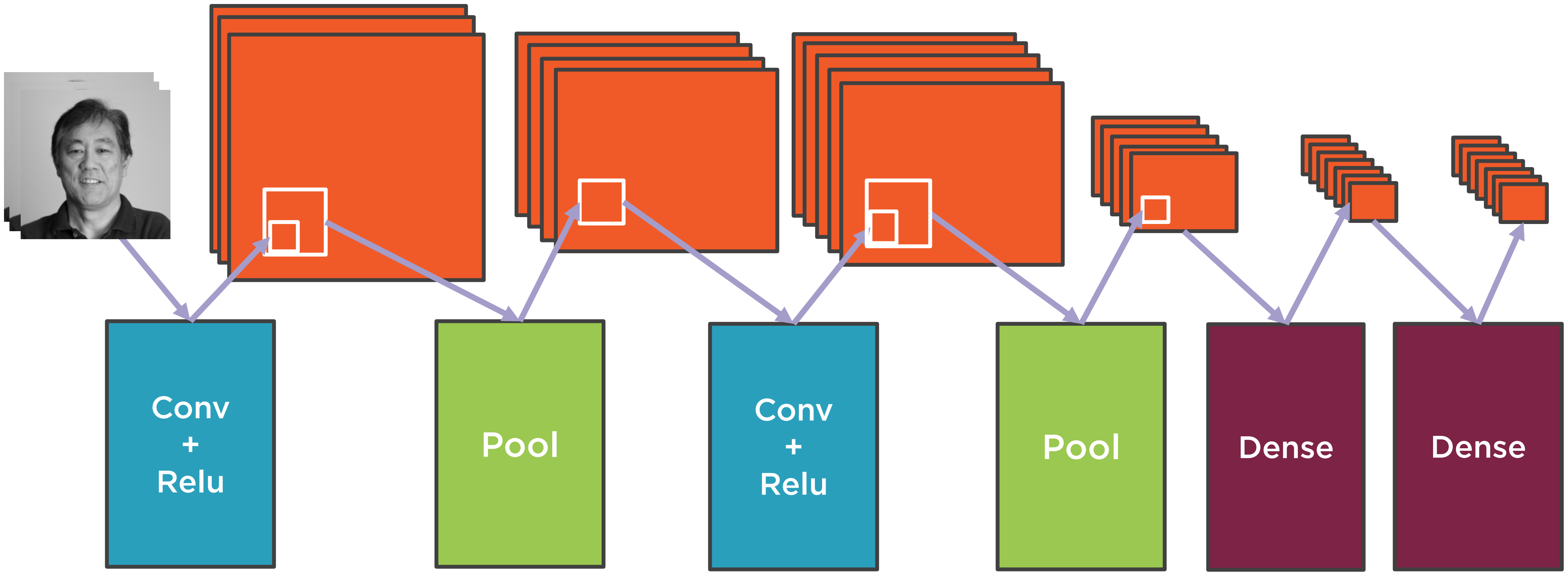
Convolutional Neural Network



Convolutional Neural Network



Convolutional Neural Network



MNIST – Modified NIST

MNIST Handwritten digits

- 60,000 training images
- 10,000 testing images

28 X 28 X 1 (grayscale)

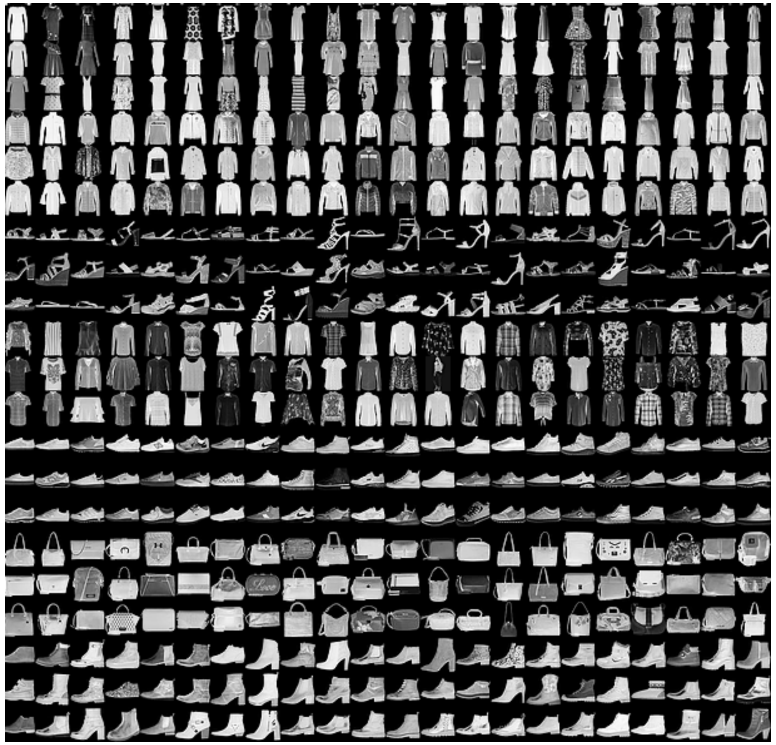
10 classes

- Digits 0 through 9

A basic CNN trained easily

Over 99% accuracy often achieved

Not a common task



Fashion-MNIST images

- 60,000 training
- 10,000 test

28 X 28 X 1 (grayscale)

10 classes

- (t-shirt, trouser, pullover, coat, sandal, skirt, sneaker, bag, and ankle boot)

Solvable by basic CNN

90% accuracy good, higher with tuning

Common task

Demo

Fashion MNIST Image Classification

Need powerful model

Don’t want to build from scratch

Transfer Learning

- Leverage existing trained model
- Tailor to our problem

Fashion MNIST

Fashion MNIST

Convolution

Pool

Concat

Dropout

Dense

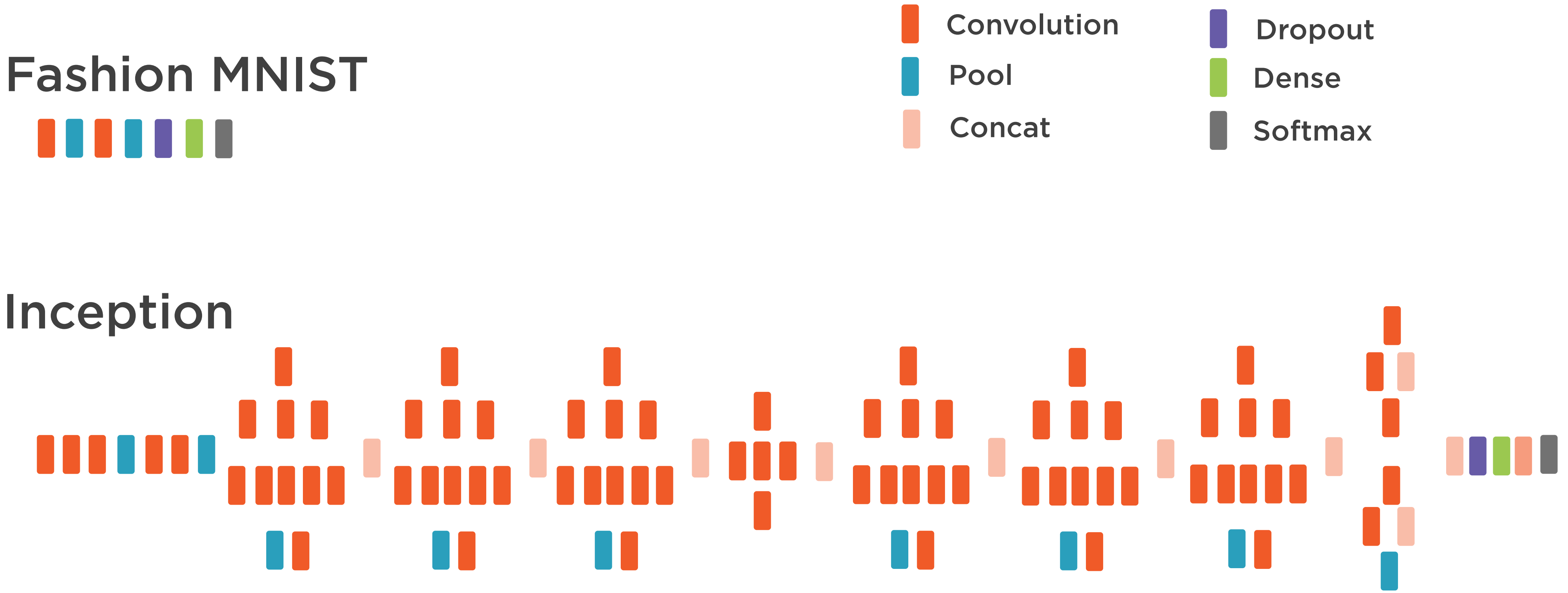
Softmax

Small (28 X 28)

Clear and well-defined

Lots of examples

Fashion MNIST vs Inception V3



Inception Training Issues

Needs lots of data	Takes several weeks
1.2 million images	Big clusters of systems

Transfer Learning Steps

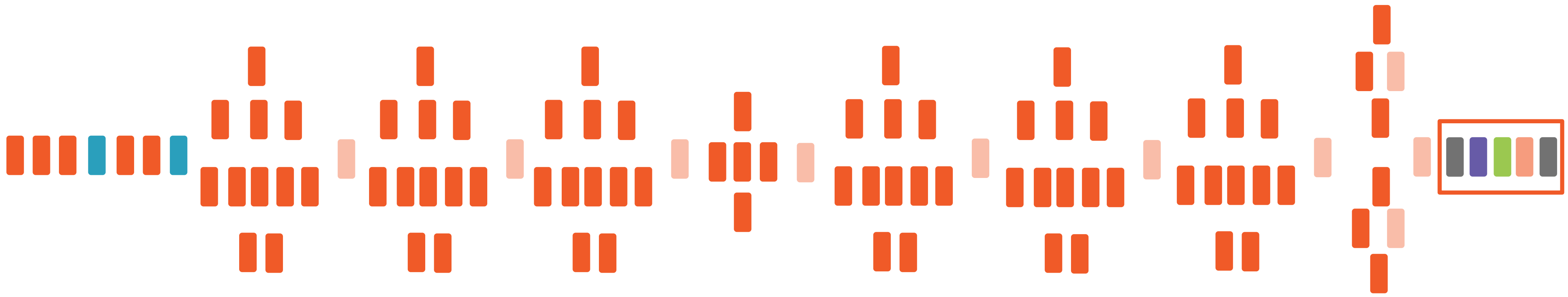
Use pre-trained model

- Leverage power of model
- Feature detection

Replace classifier with our classifier

Train classifier on our classes

Transfer Learning Retrain



Demo

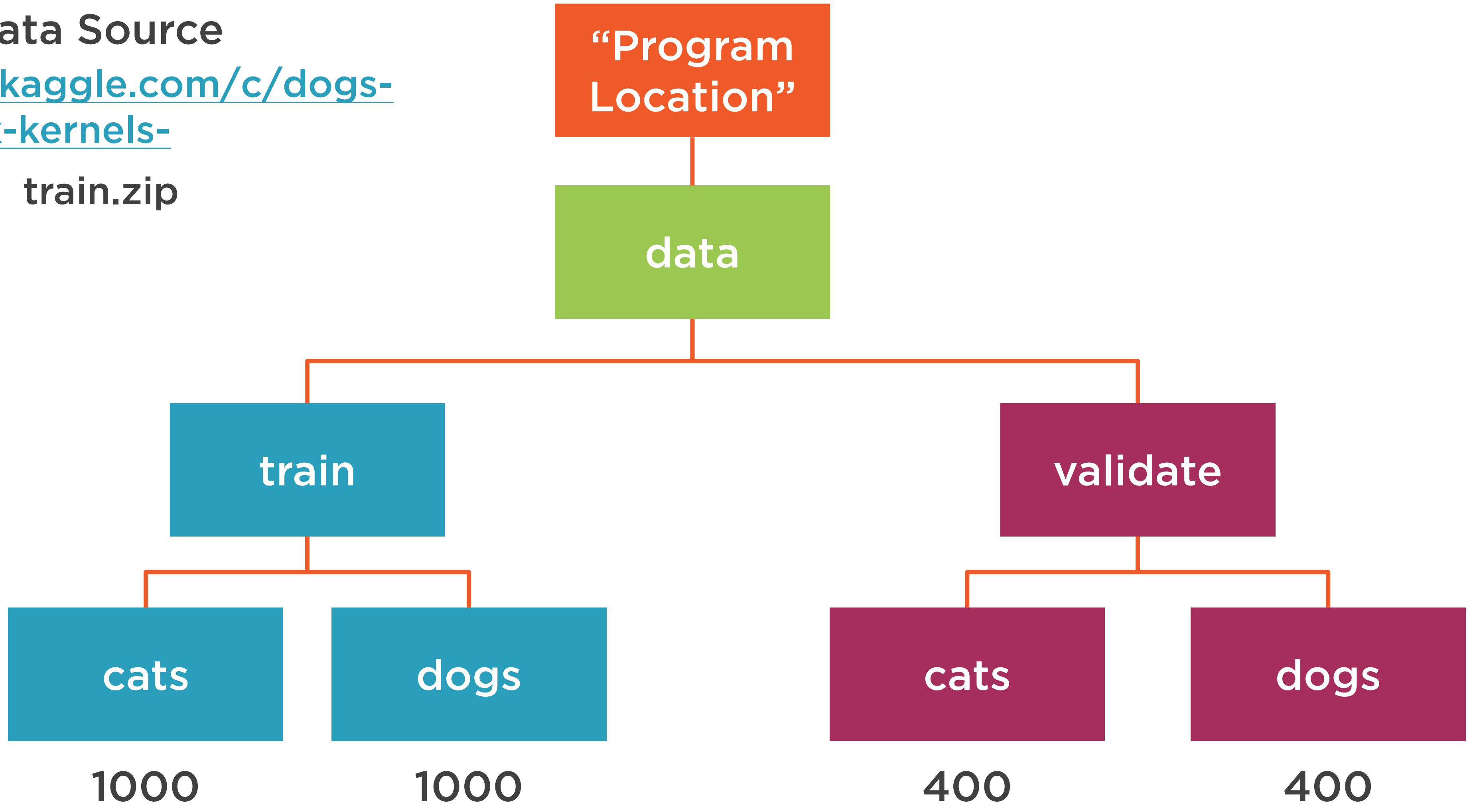
Transfer Learning

Data Folder Structure

Data Source

https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data

train.zip



More Than Static Images

Video

Signal Processing

Natural Language Processing

Combined with RNNs

Summary

Reviewed image data issues

- Lots of weights
- Translation Invariance

CNNs solved issues

CNNs build Feature Maps

Built FashionMNIST model from scratch

Utilized Transfer Learning

CNNs easy with Keras