Creating Neural Networks in TensorFlow

Jerry Kurata
CONSULTANT

@jerrykur www.insteptech.com

Training Steps

Prepared Data

Inference

Loss Measurement

Optimize to Minimize Loss

Overview

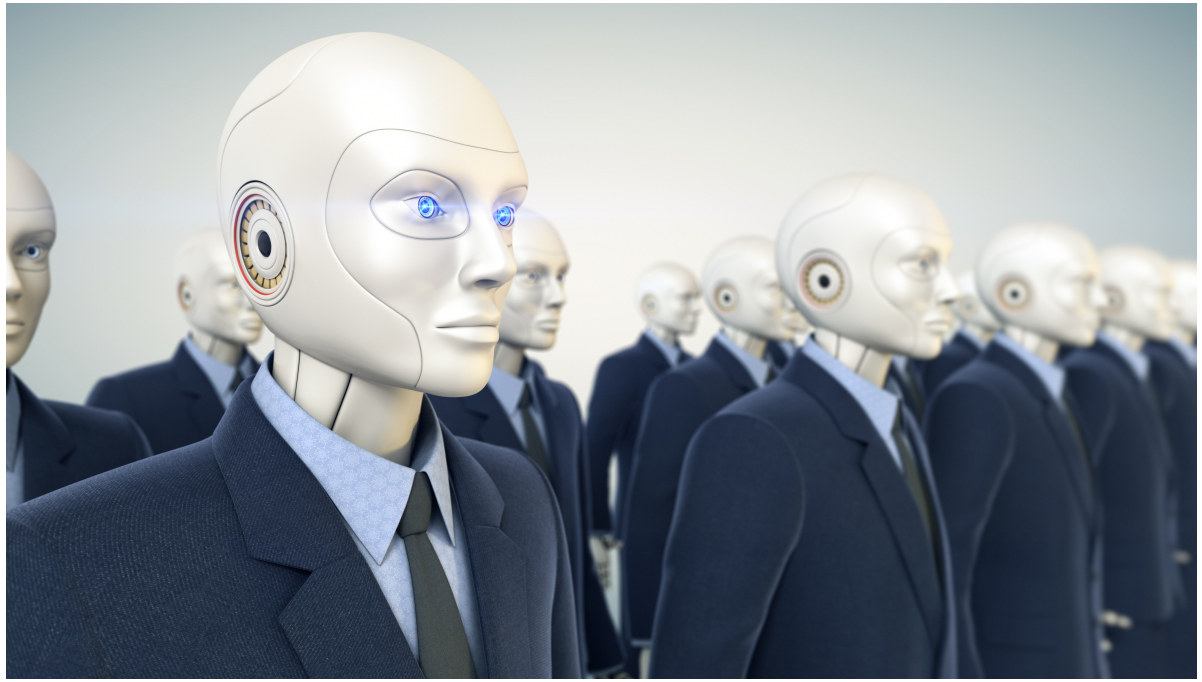
Introduction to Neural Networks

Building a simple Neural Network

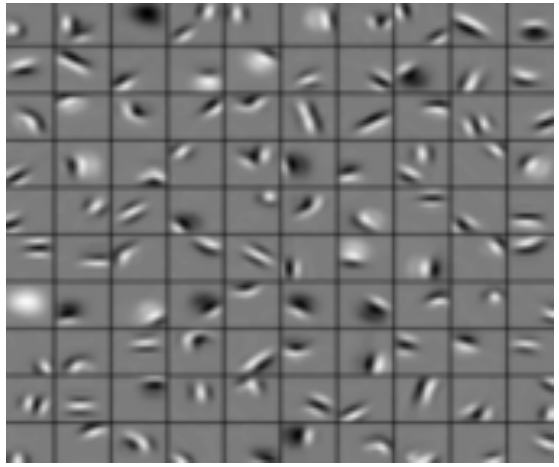
Creating Deep Neural Networks

TensorFlow makes Neural Networks simple

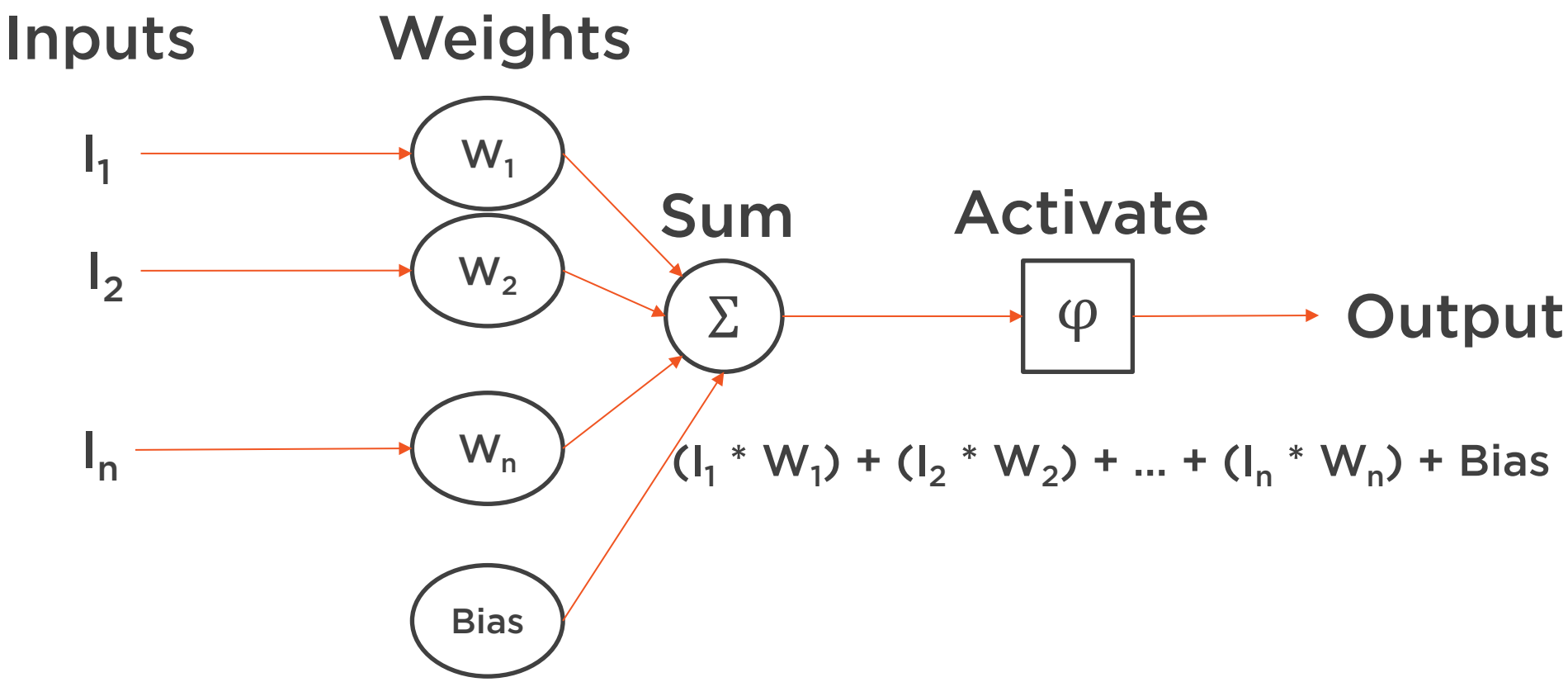
Neural Networks are



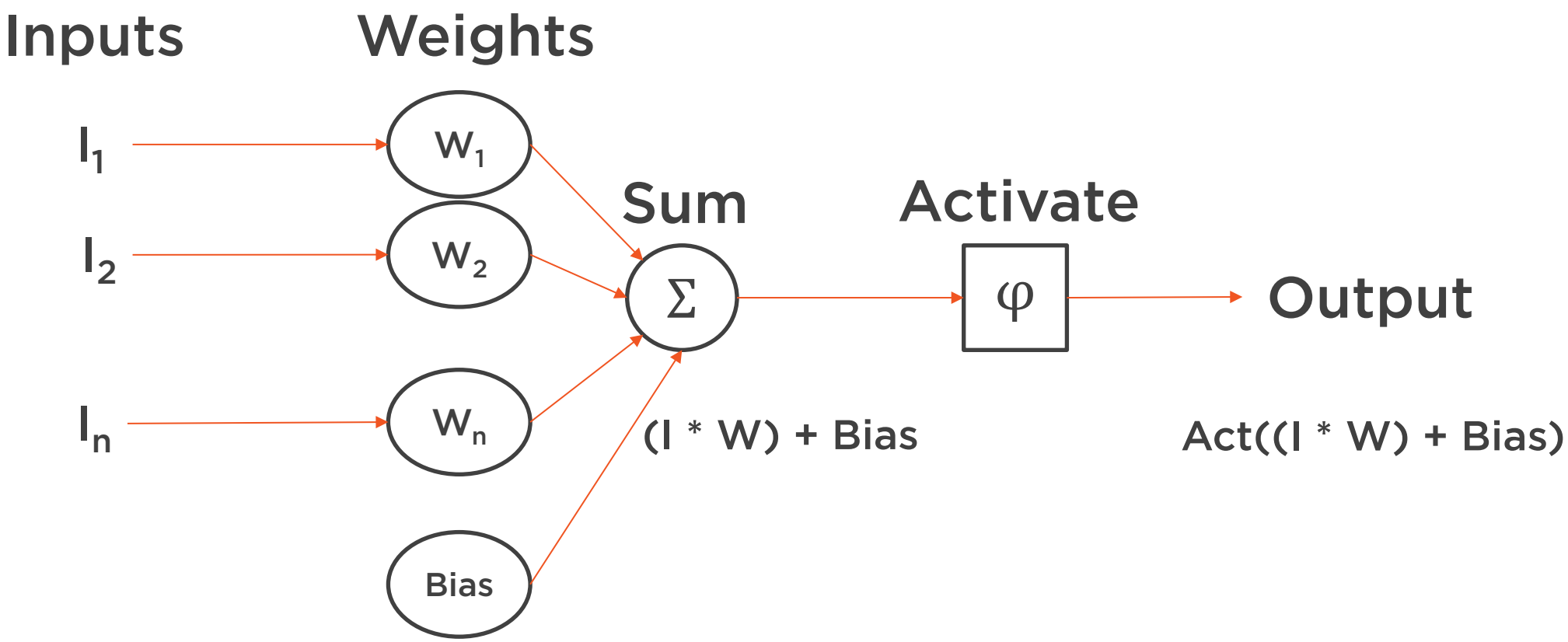
Neural Network Learning



Neuron Architecture



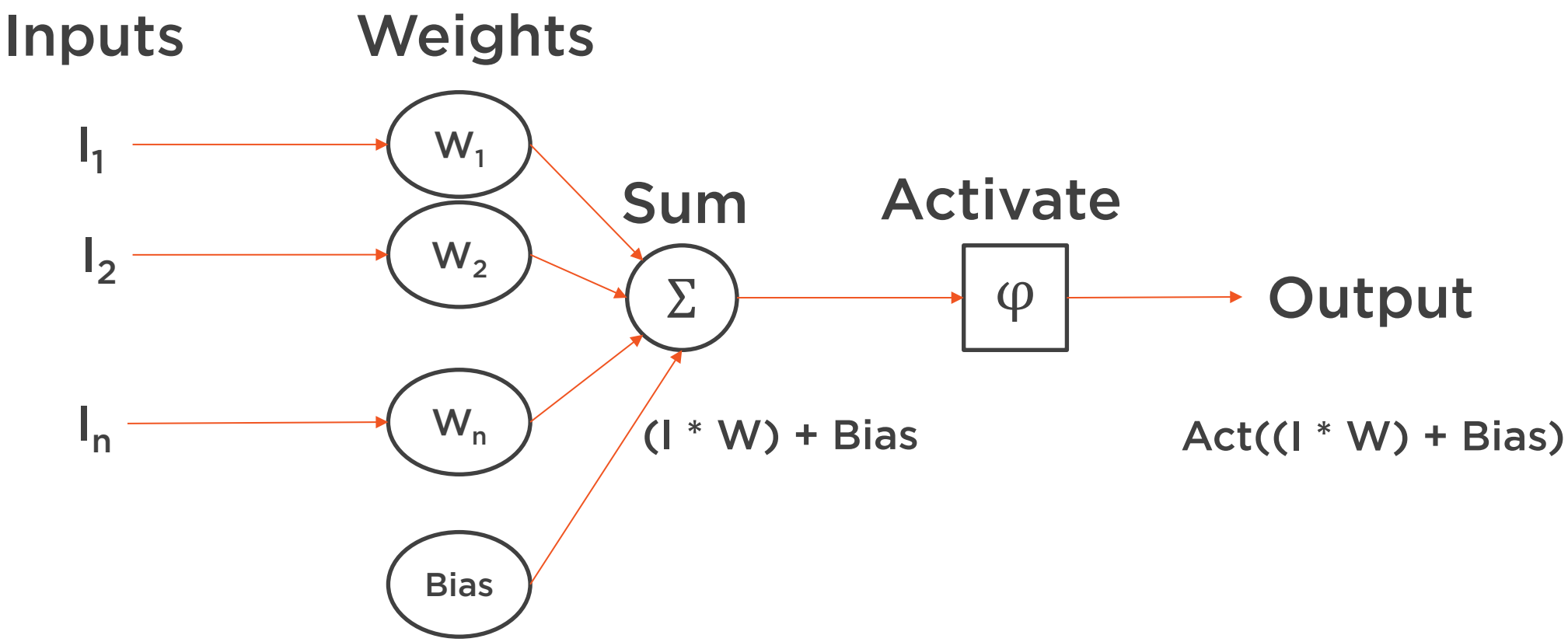
Neuron Architecture



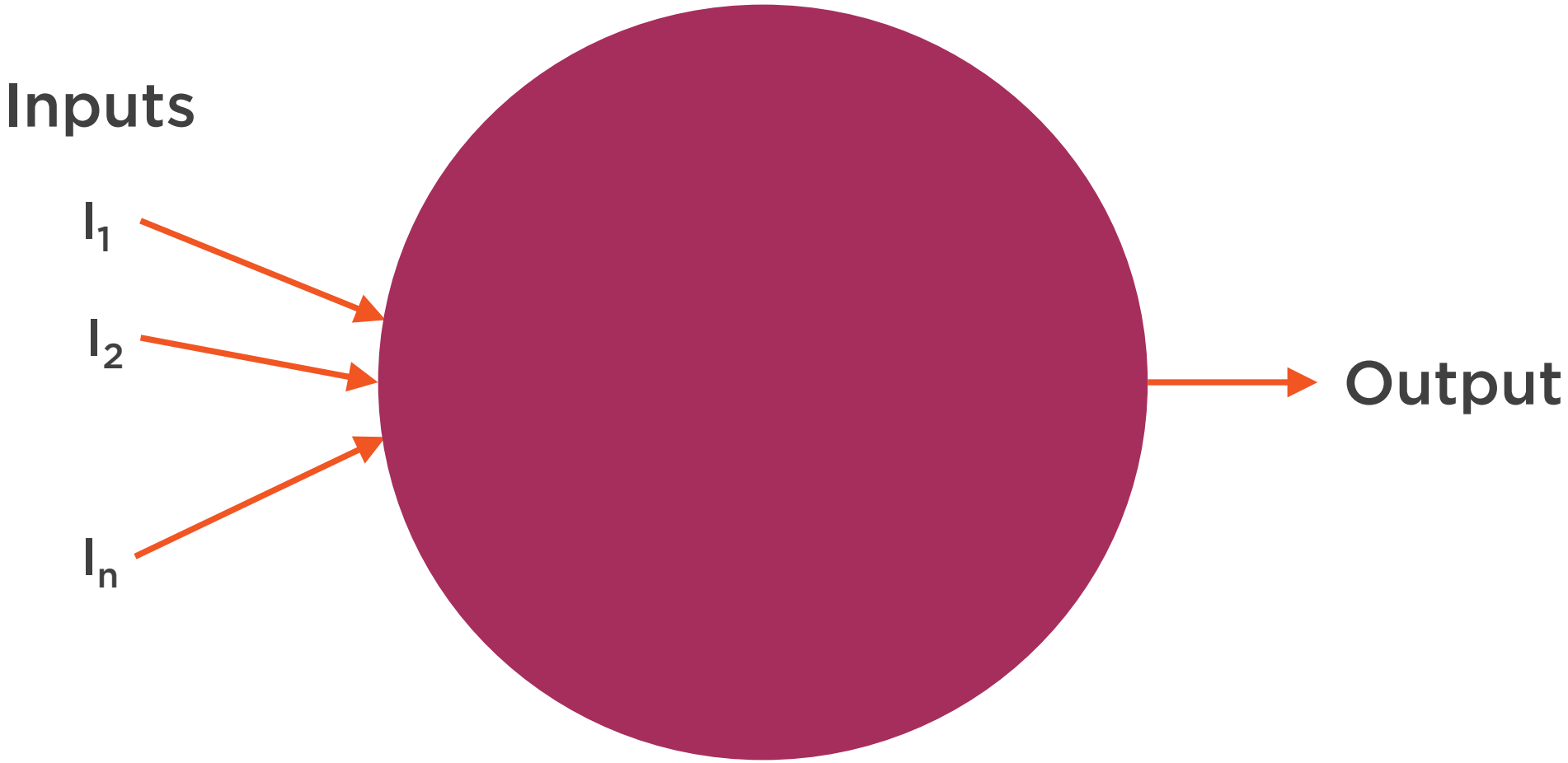
Linear Regression Example Revisited

```
tf_price_pred = tf.add(tf.multiply(tf_size_factor,
                                   tf_house_size), tf_price_offset)


tf_cost = tf.reduce_sum(tf.pow(tf_price_pred-tf_price, 2))
                  /(2*num_train_samples)


optimizer = tf.train.GradientDescentOptimizer(learning_rate)
                  .minimize(tf_cost)
```

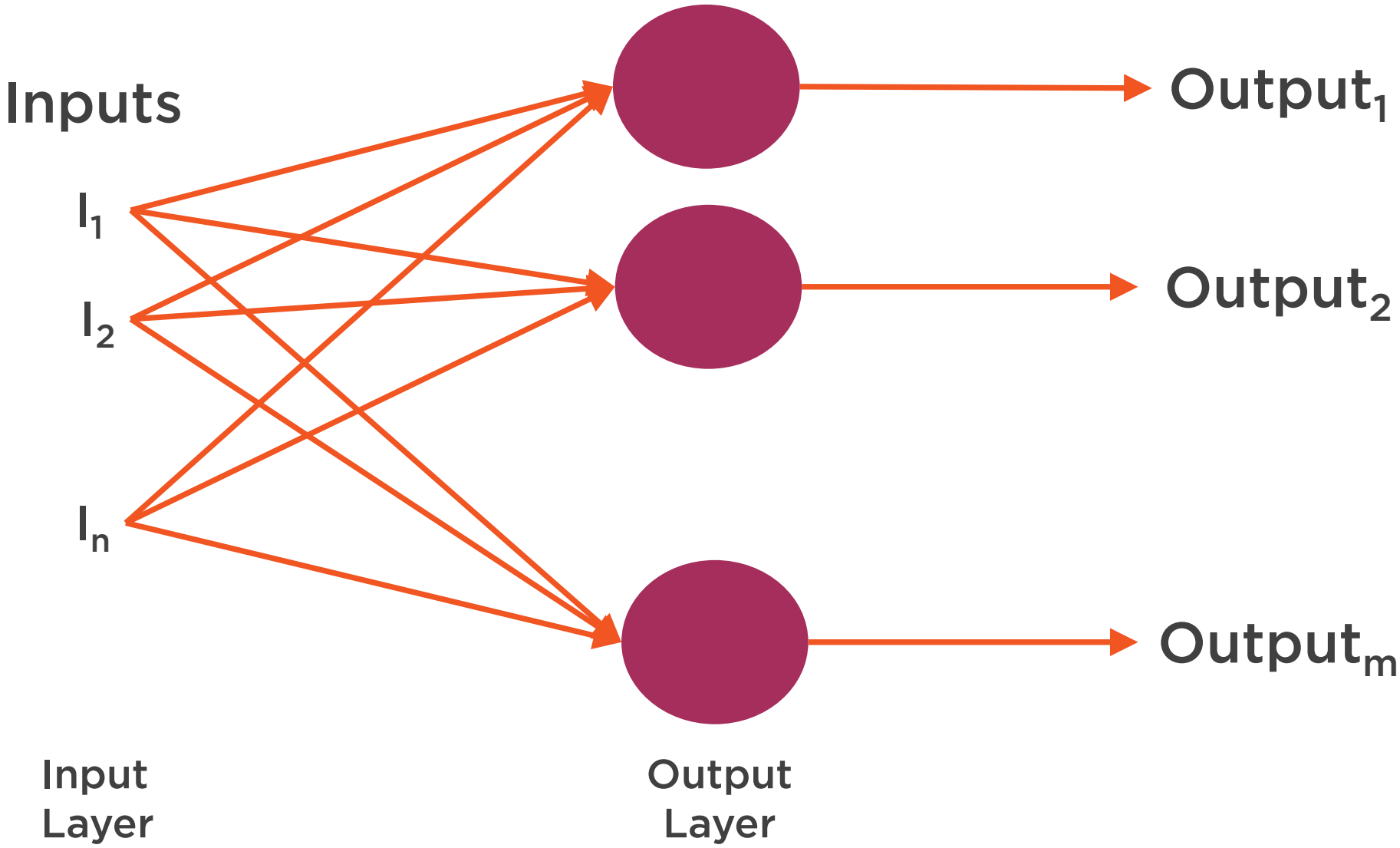
Neuron Architecture



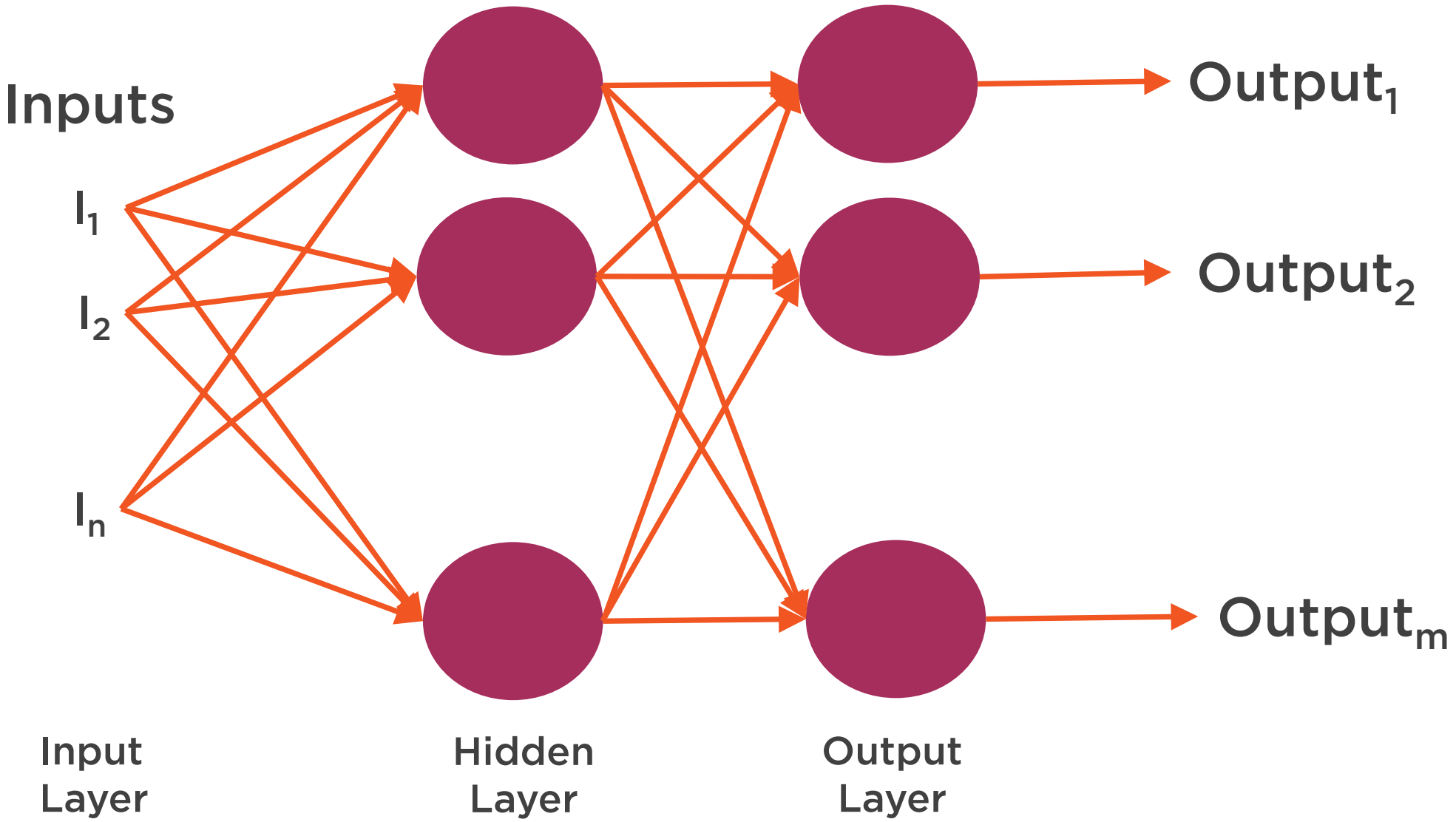
Neuron



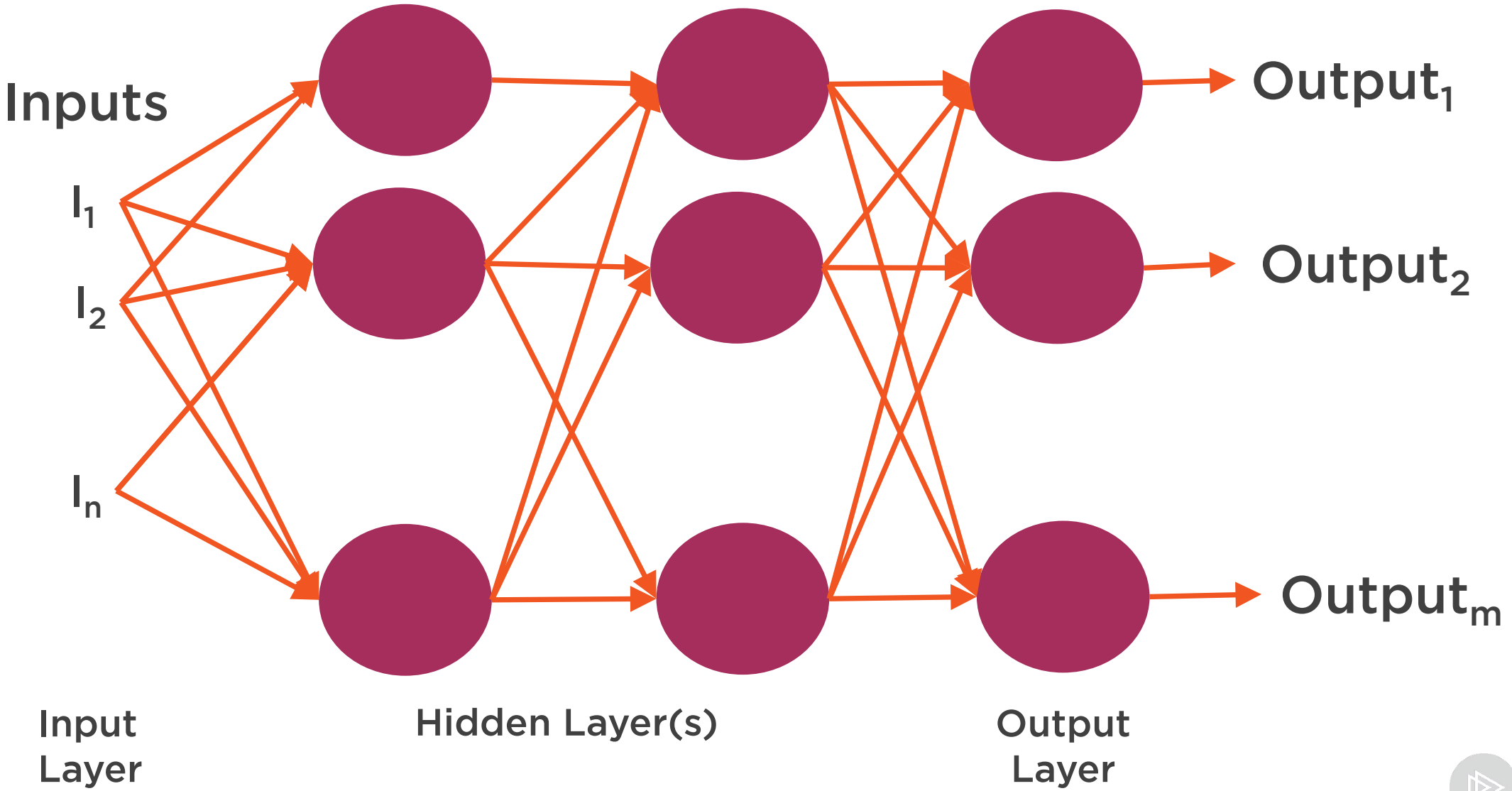
Neural Network Layers



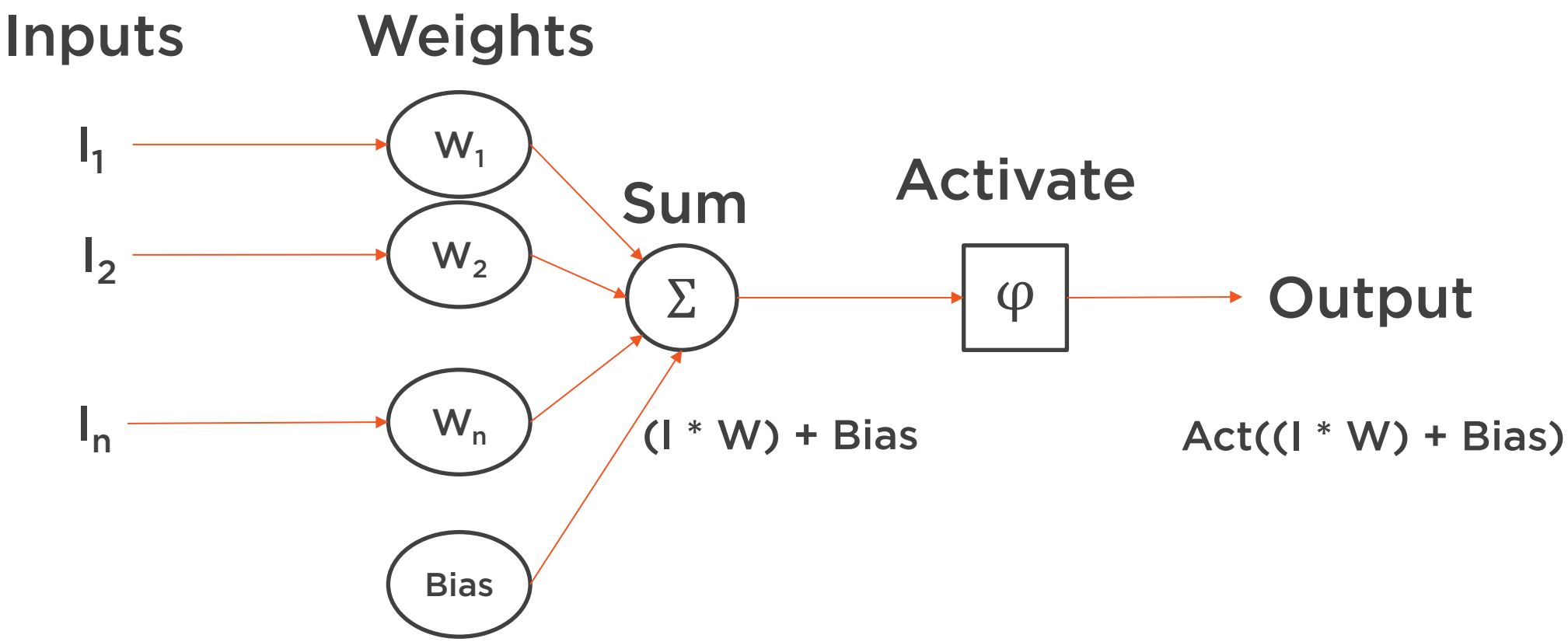
Neural Network Layers



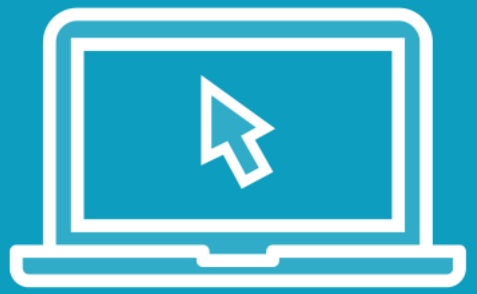
Deep Neural Network Layers



Neuron Architecture



Simple Neural Network



Handwritten digit recognition

MNIST dataset

yann.lecun.com/exdb/mnist

Classic testing set

MNIST Data

70,000 data points

- 55,000 training
- 10,000 test
- 5,000 validation

Each data point contains

- 28 X 28 grayscale image
- Label – the digit, 0-9

Helper routines on TensorFlow site

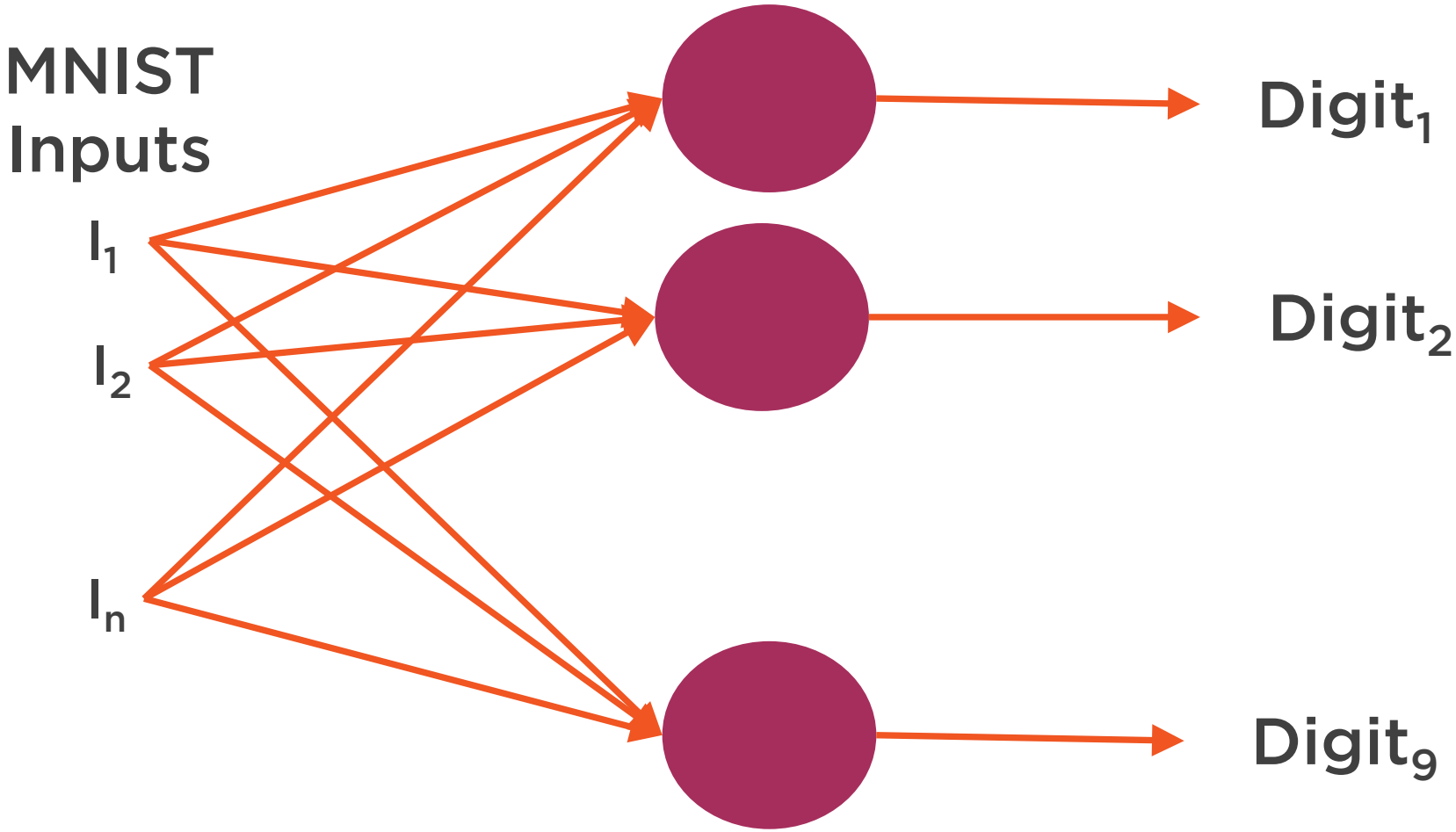
Training a Neural Network with TensorFlow

Concept	Implementation
Prepared Data	MNIST Data
Inference	(x * weight + bias) -> activation
Loss Measurement	Cross Entropy
Optimize to Minimize Loss	Gradient Descent Optimizer

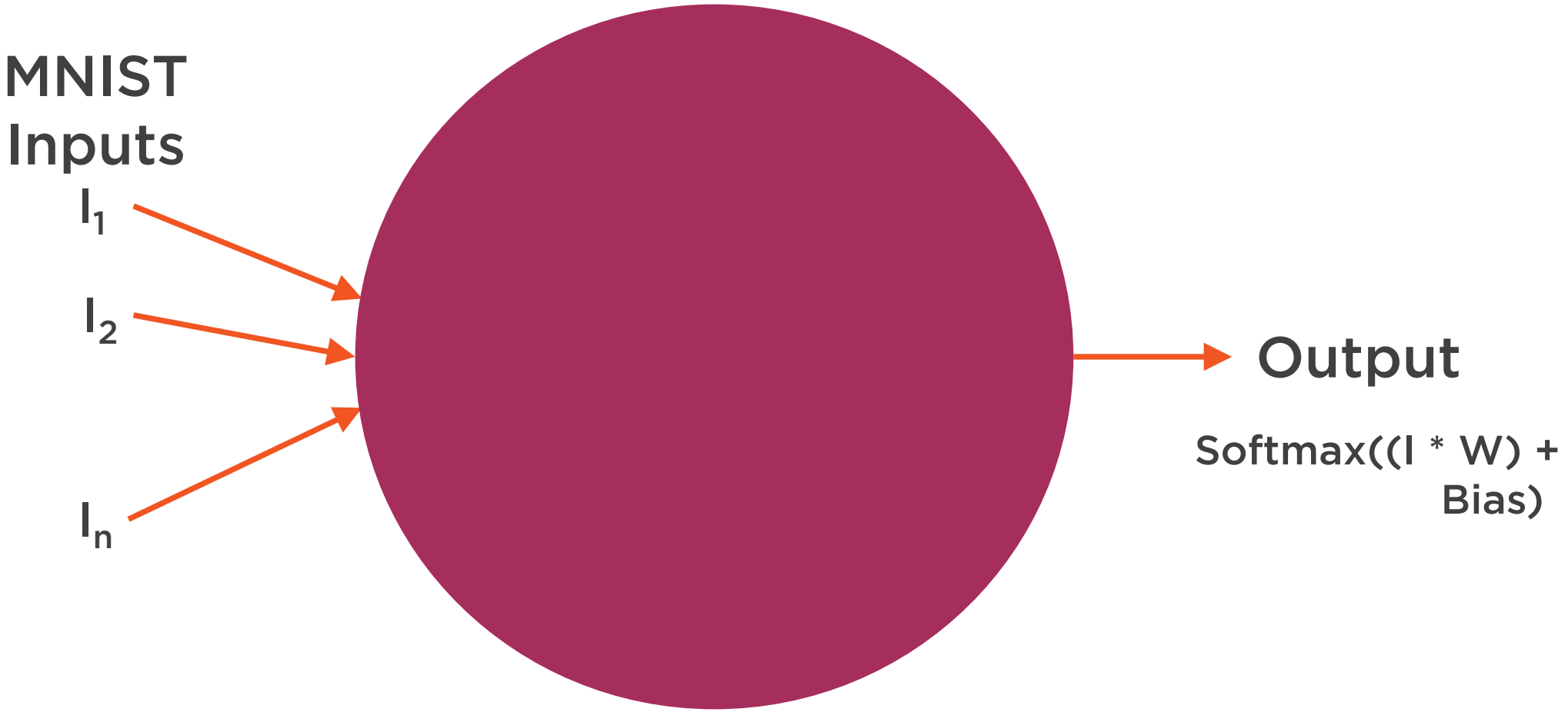
Create MNIST_Basic.py

DO NOT SHOW SLIDE

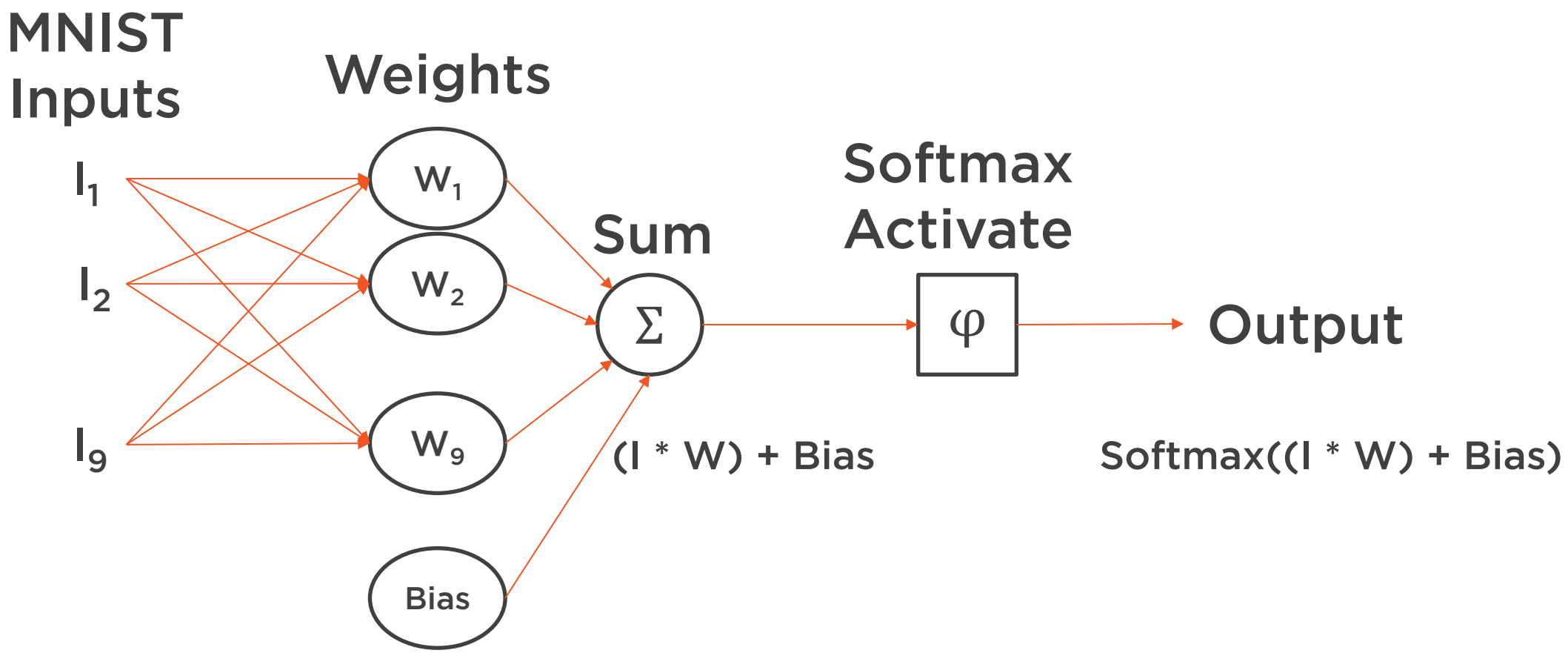
MNIST Neural Network



MNIST Neuron



MNIST Neuron Architecture

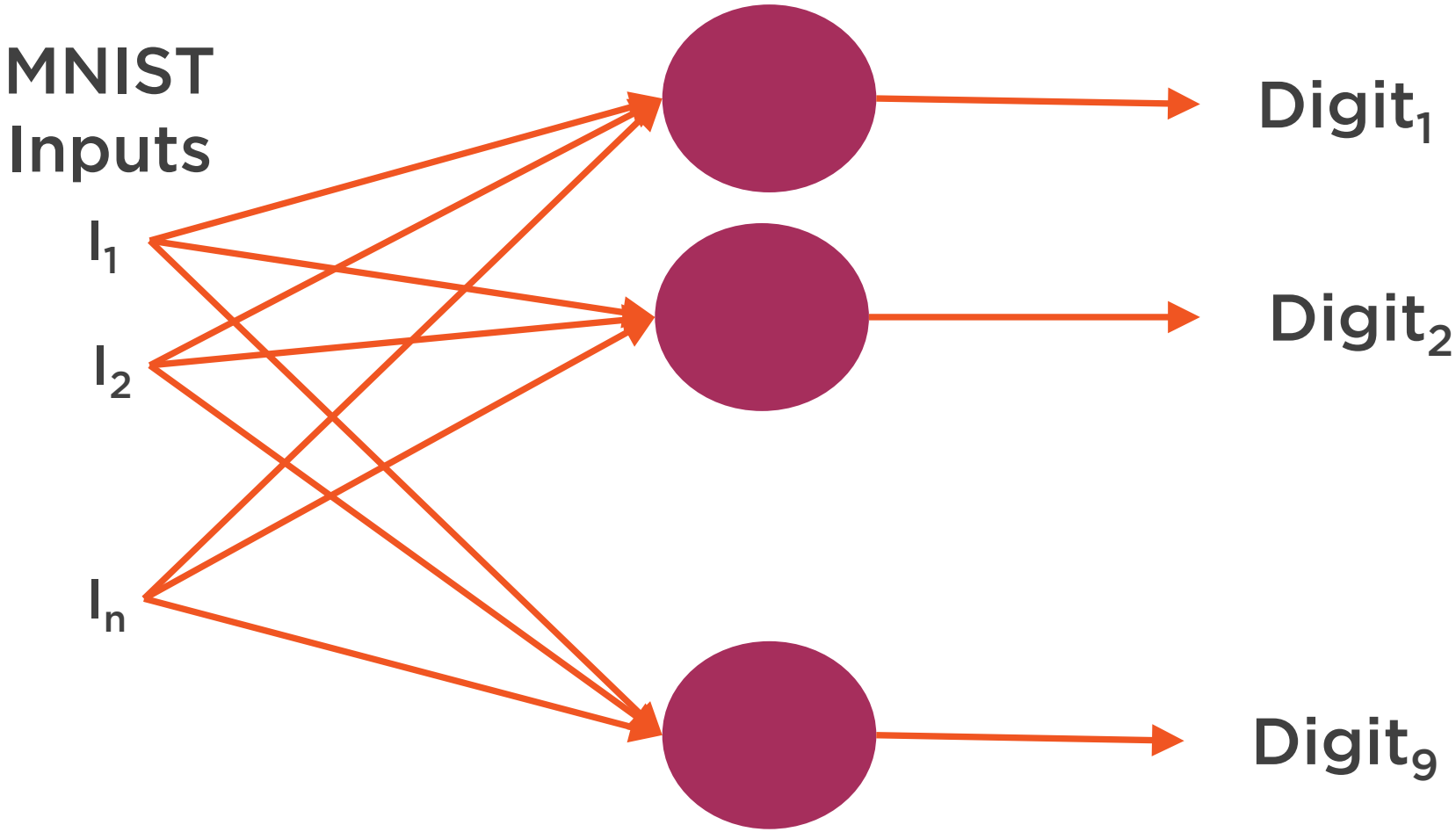


Improving Performance by Going Deep

Deeper networks can learn more details

Simple to do with TensorFlow

Simple MNIST Neural Network



Deepening the Network

Consider data

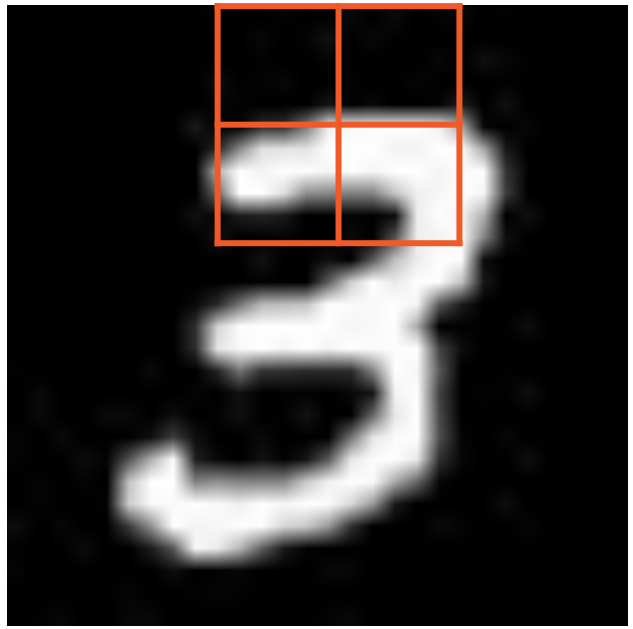
- We have images!

Insert Convolution Network layers

- Inspects subsets of image
- Learns features of image

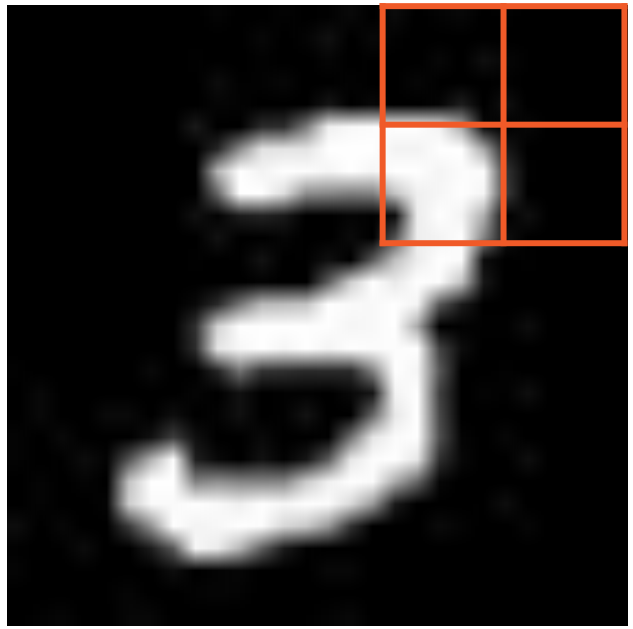
Convolution Layer

Convolution Layer



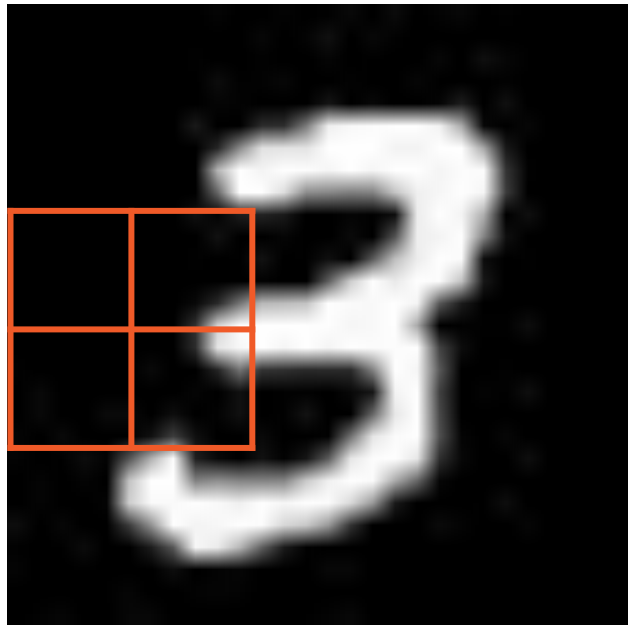
.0	.0
.0	.2

Convolution Layer



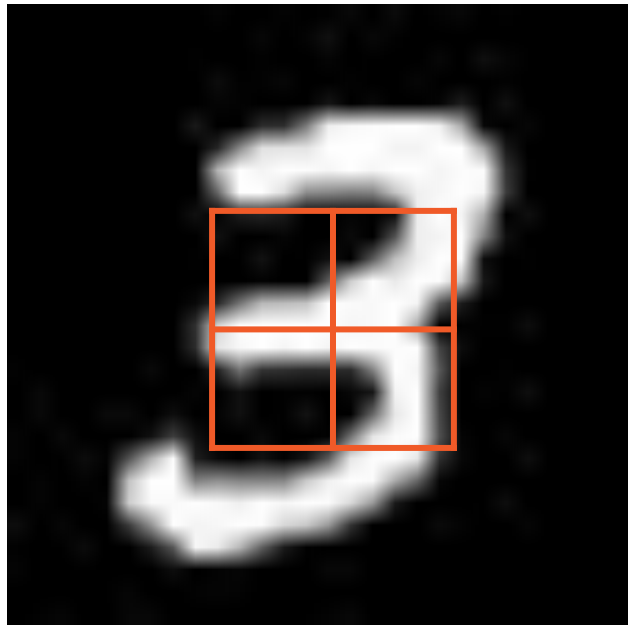
.0	.0	.1	.2
.0	.2	.7	.9

Convolution Layer



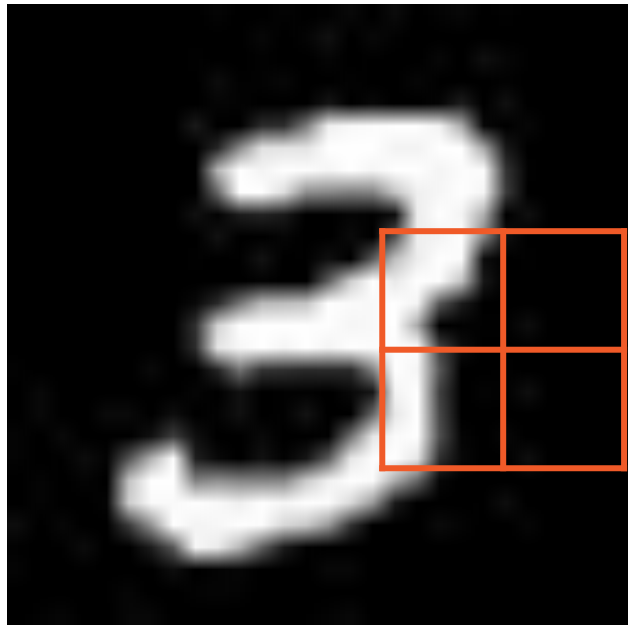
.0	.0	.1	.2	.1	.0
.0	.2	.7	.9	.9	.0

Convolution Layer



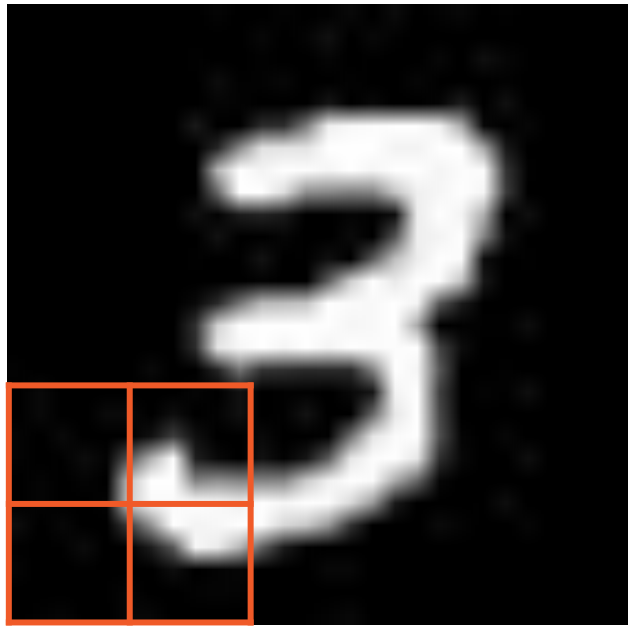
.0	.0	.1	.2	.1	.0
.0	.2	.7	.9	.9	.0
.0	.2				
.0	.2				

Convolution Layer



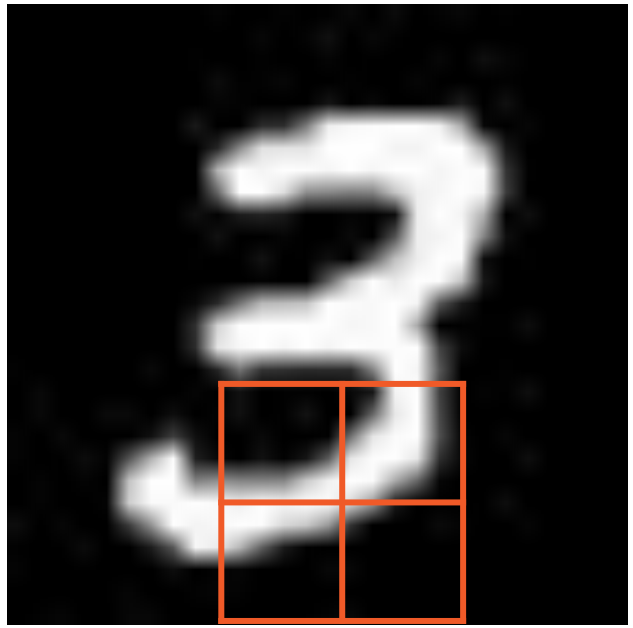
.0	.0	.1	.2	.1	.0
.0	.2	.7	.9	.9	.0
.0	.2	.4	.5	.8	.0
.0	.2	.3	.7	.5	.0

Convolution Layer



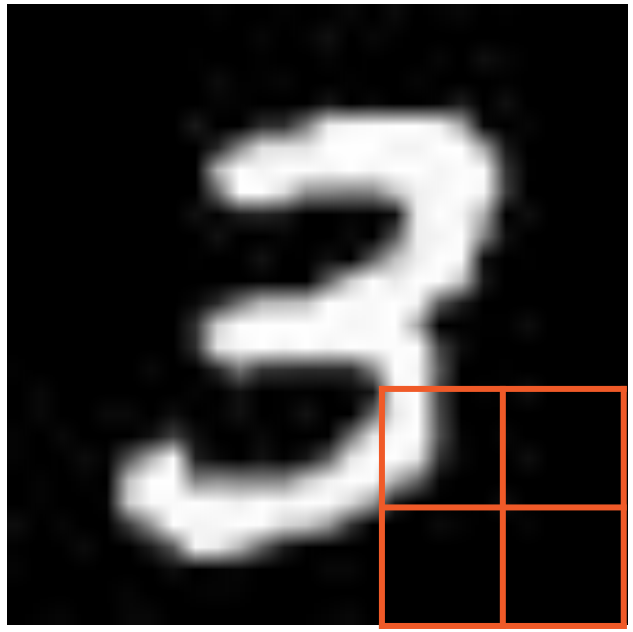
.0	.0	.1	.2	.1	.0
.0	.2	.7	.9	.9	.0
.0	.2	.4	.5	.8	.0
.0	.2	.3	.7	.5	.0

Convolution Layer



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

Convolution Layer



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

Convolution Layer

Sample 1

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

Convolution Layer

Sample 2

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

Convolution Layer

Sample 3

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

Convolution Layer

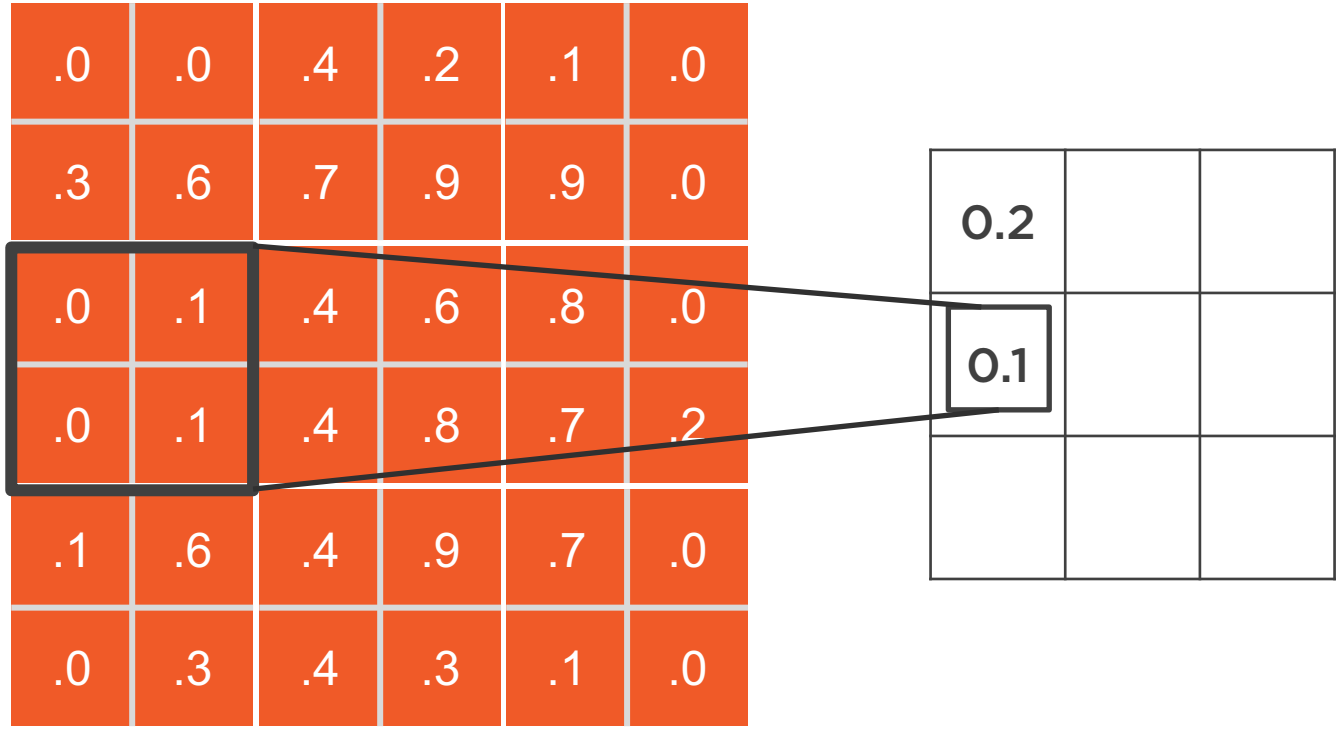
.0	.0	.4	.2	.1	.0
.3	.6	.7	.9	.9	.0
.0	.1	.4	.6	.8	.0
.0	.1	.4	.8	.7	.2
.1	.6	.4	.9	.7	.0
.0	.3	.4	.3	.1	.0

Convolution Layer + Pool Layer

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

0.2		

Convolution Layer + Pool Layer



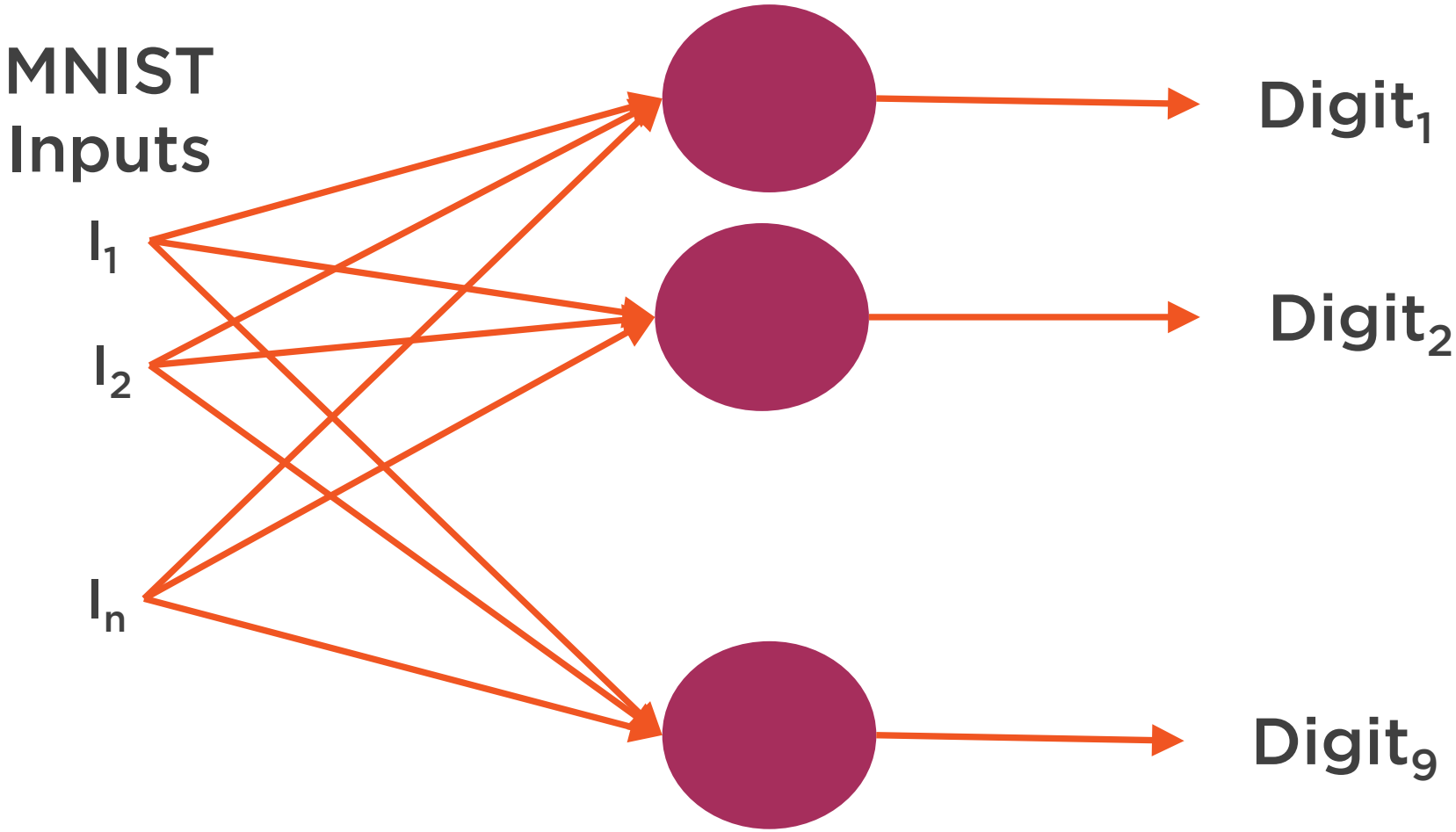
Convolution Layer + Pool Layer

Convolution Layer + Pool Layer

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

0.2	0.6	0.3
0.1	0.5	0.4
0.2	0.5	0.3

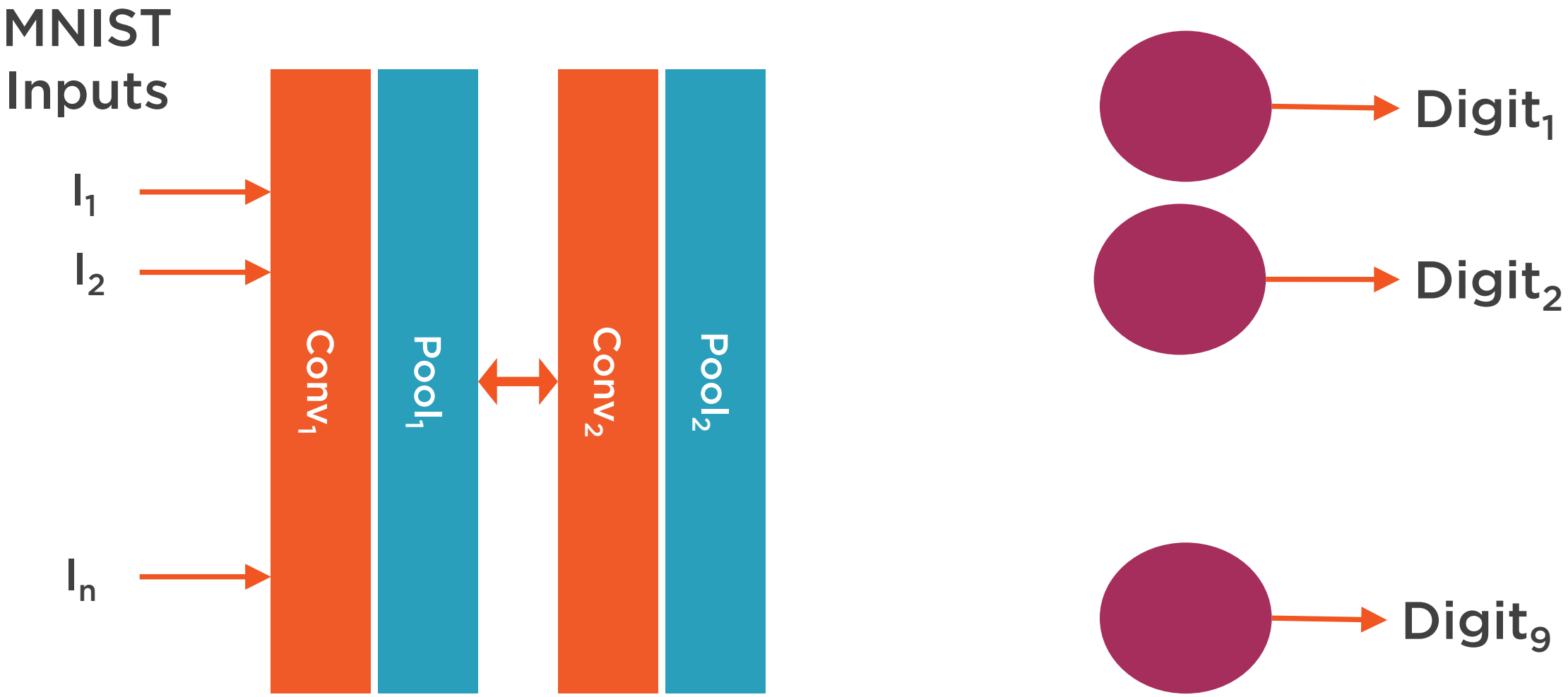
Simple MNIST Neural Network



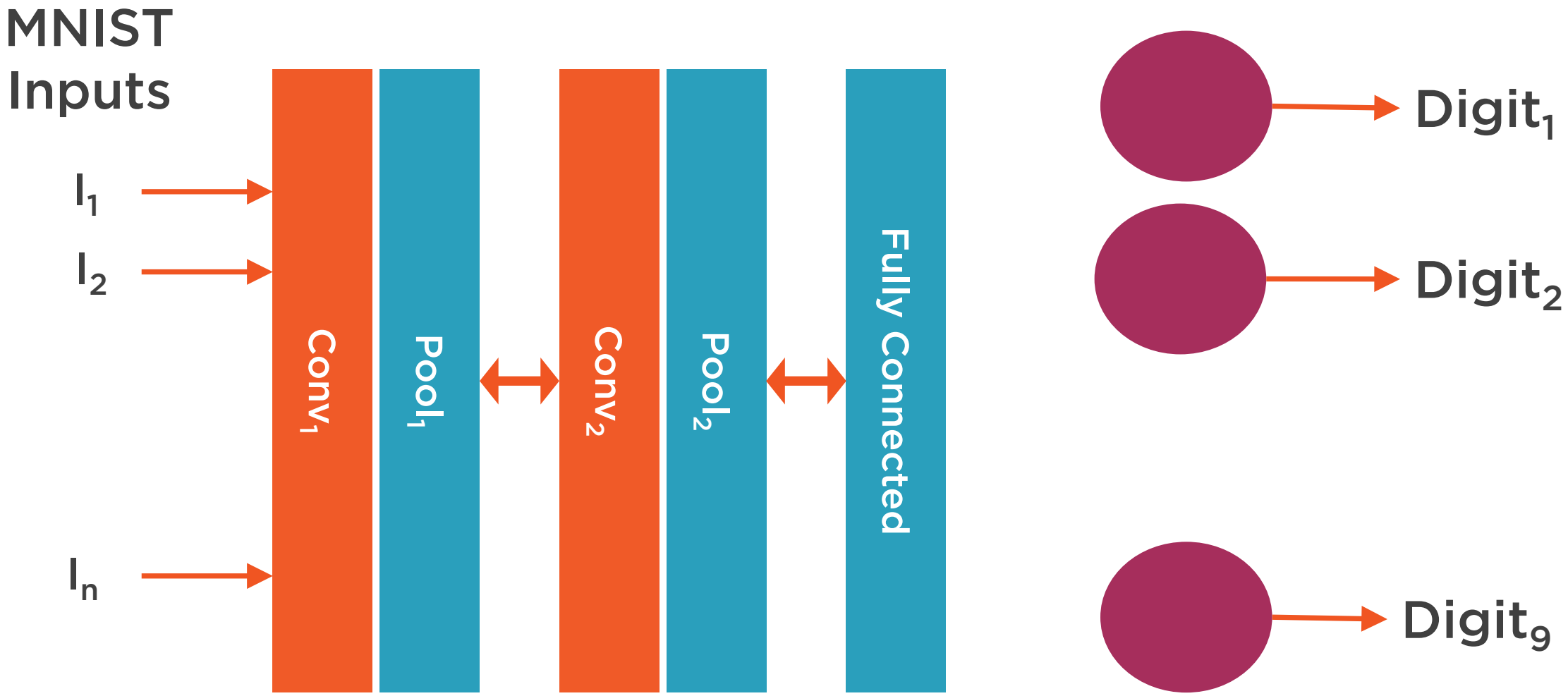
Deep MNIST Neural Network

Deep MNIST Neural Network

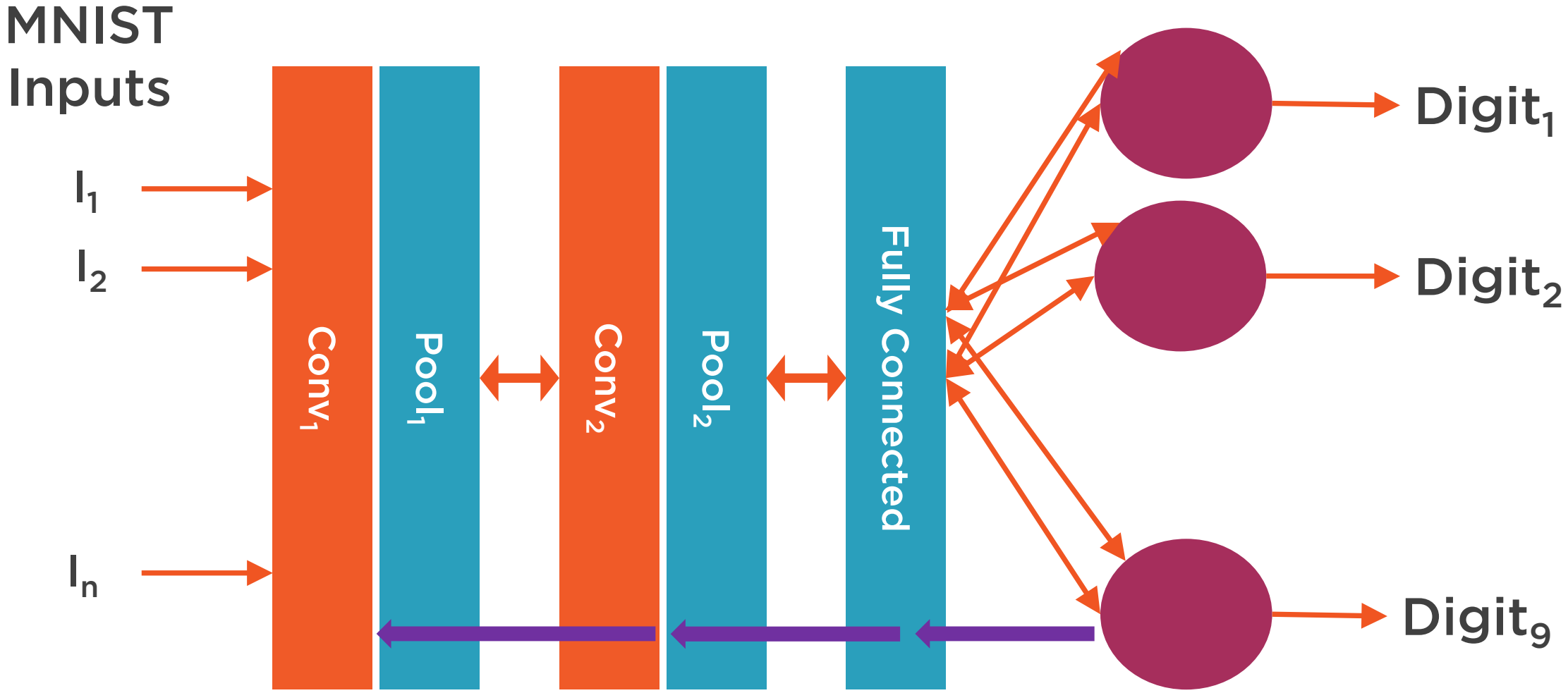
Deep MNIST Neural Network



Deep MNIST Neural Network

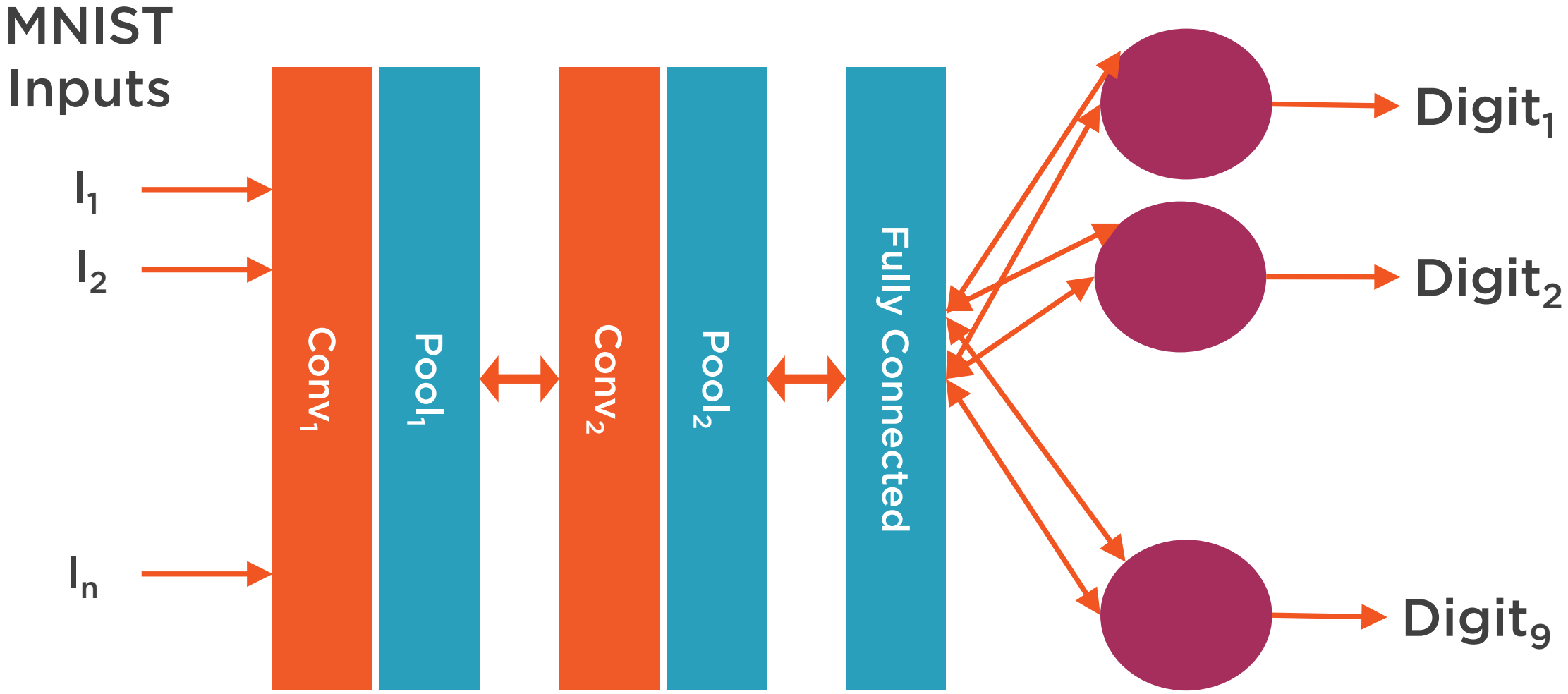


Deep MNIST Neural Network



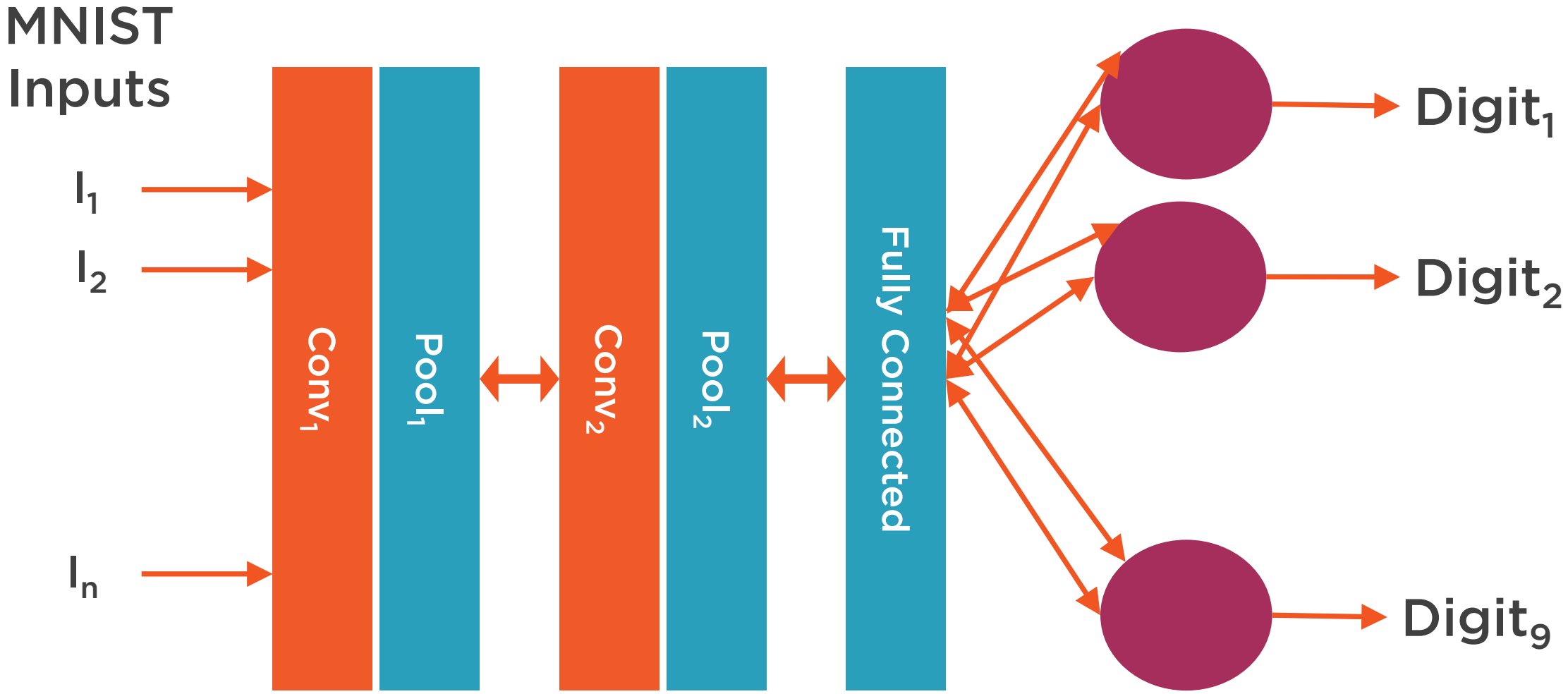
Back Propagation => “BackProp”

Deep MNIST Neural Network



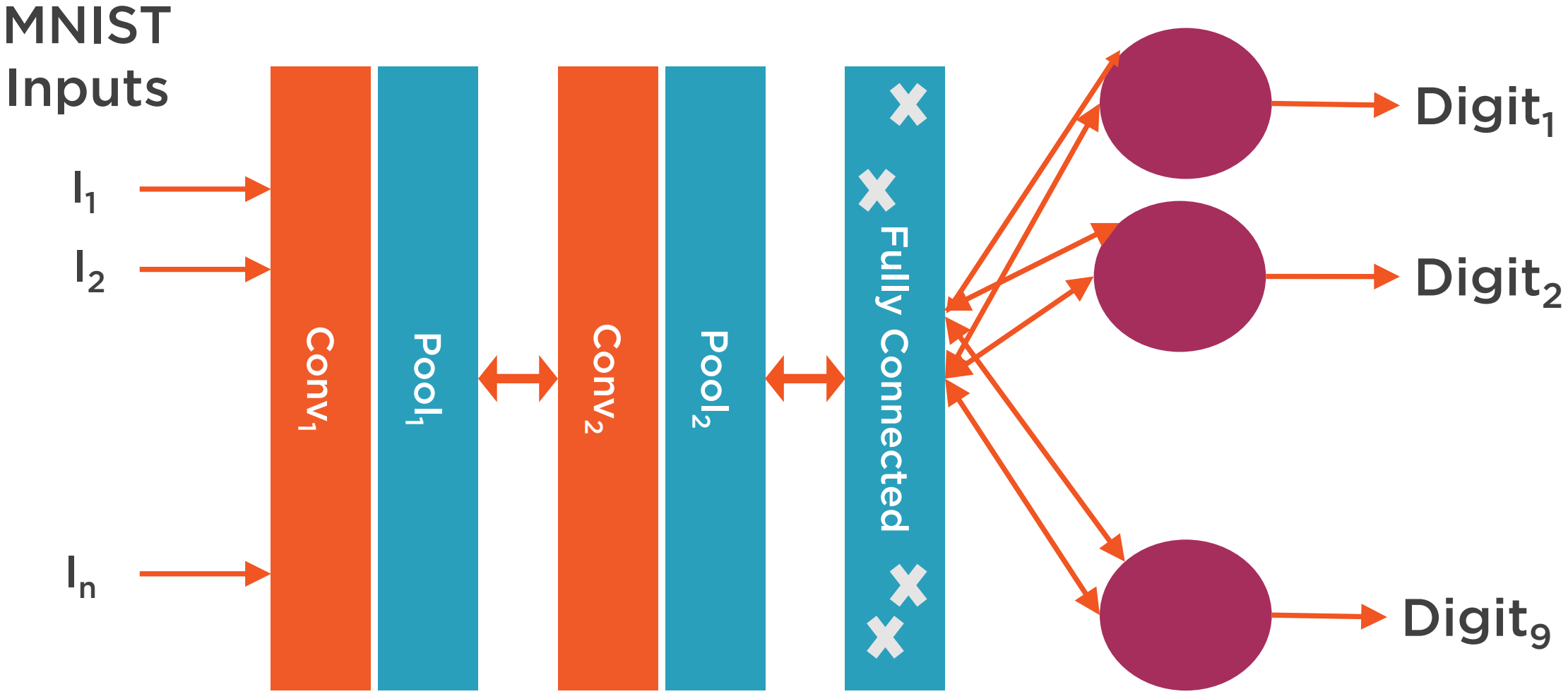
Forward Propagation

Deep MNIST Neural Network



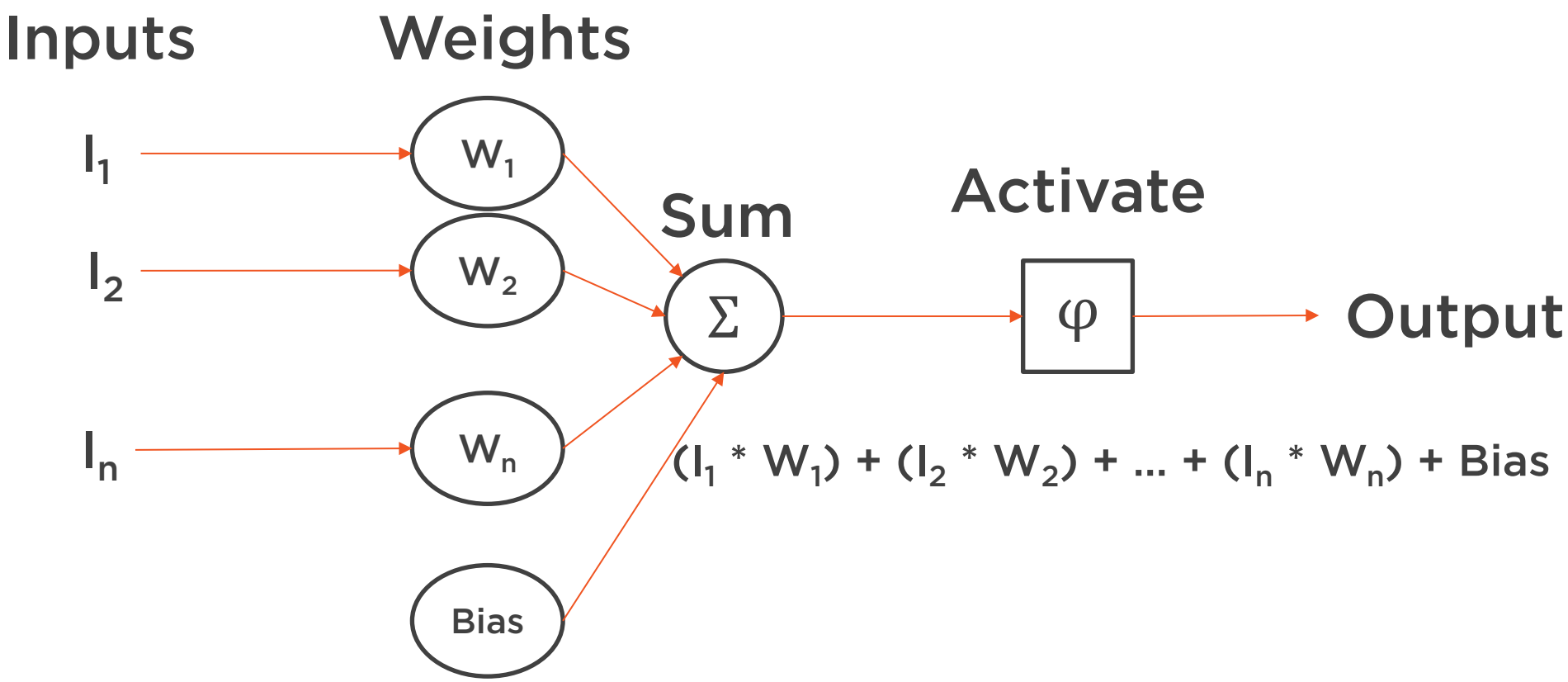
Backward Propagation

Deep MNIST Neural Network

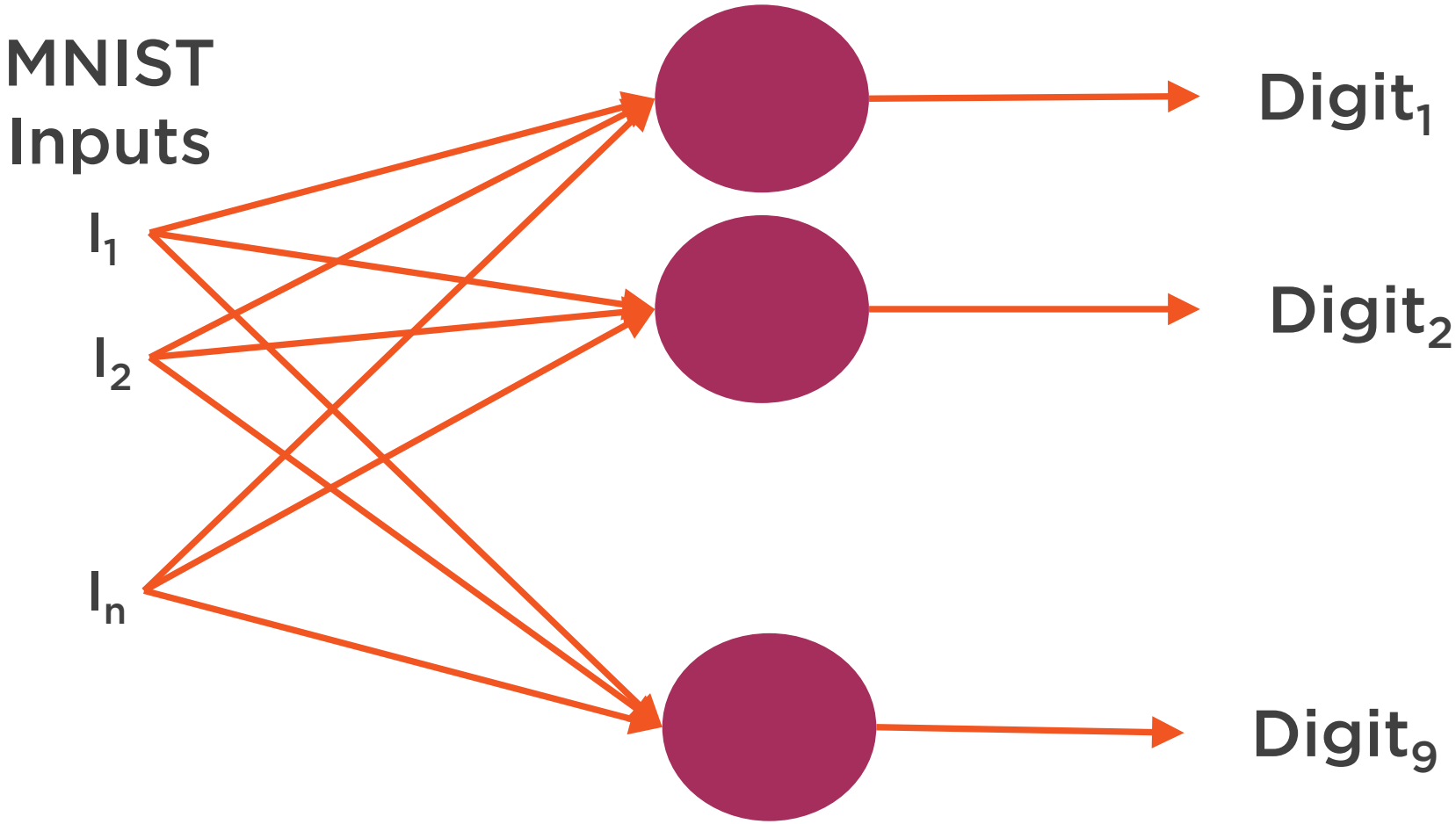


Create Deep MINST

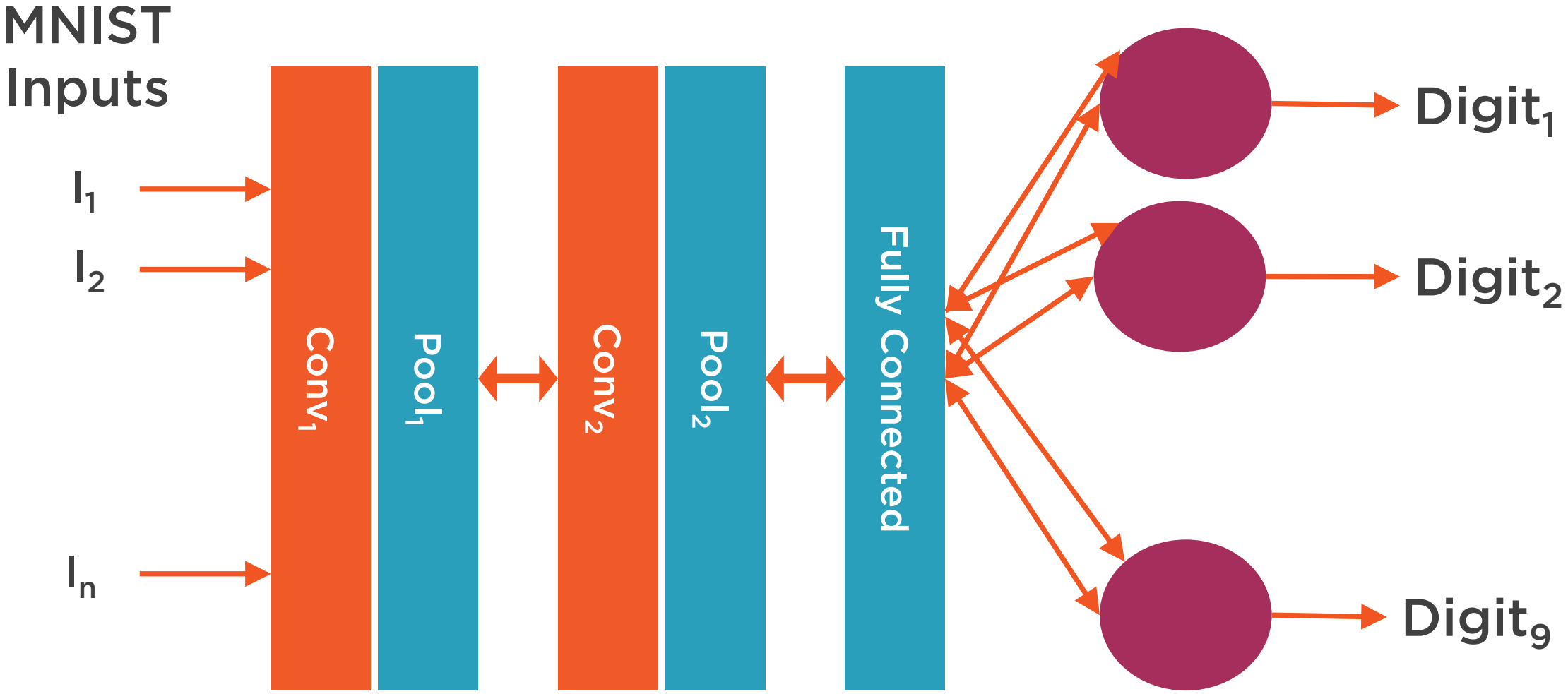
Neuron Architecture



MNIST Neural Network



Deep MNIST Neural Network



Neural Network with TensorFlow

Concept	Implementation
Prepared Data	MNIST data and reshape as required
Inference	Matmul(Weight, x) + bias for entire NN
Loss Measurement	Cross Entropy
Optimize to Minimize Loss	Gradient Decent Optimizer