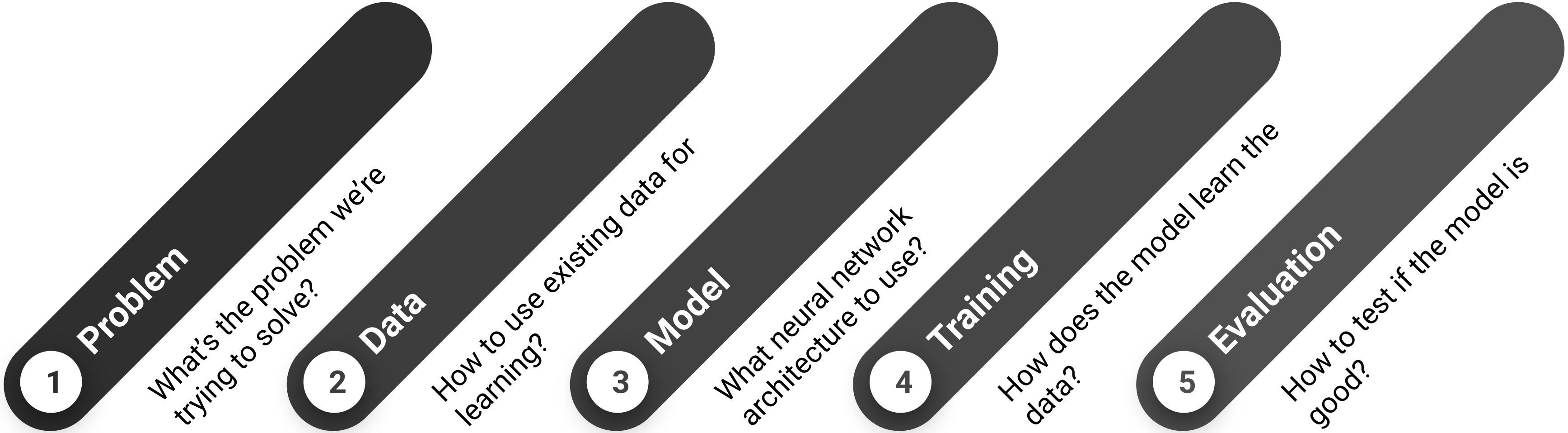
Digits Classification using MLP

UTMIST Study Group

Problem Solving Process Overview



1 **Problem**

What's the problem we're trying to solve?

2 **Data**

How to use existing data for learning?

3 **Model**

What neural network architecture to use?

4 **Training**

How does the model learn the data?

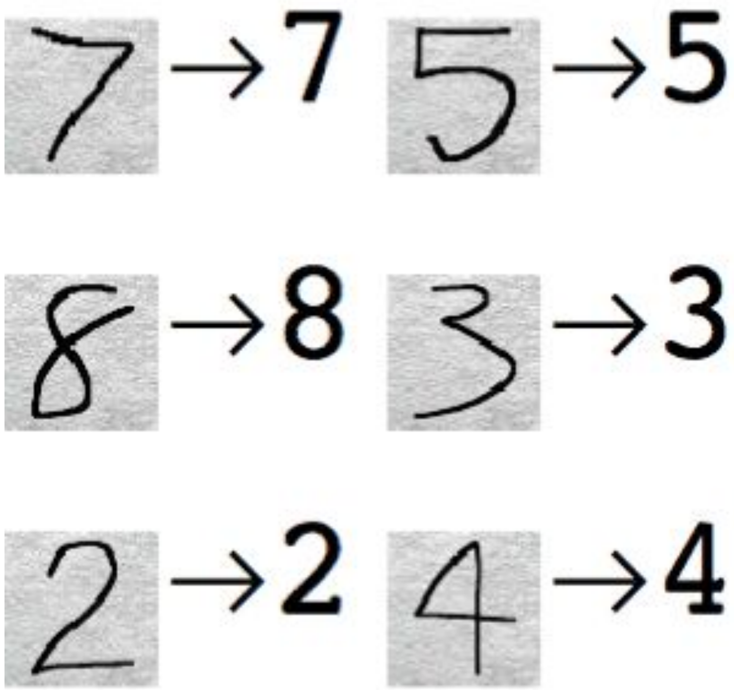
5 **Evaluation**

How to test if the model is good?

Problem Statement

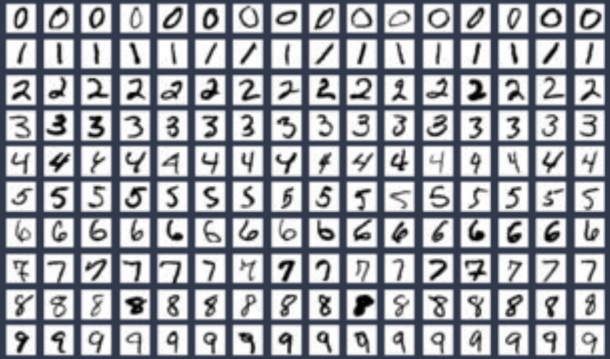
We'd like to recognize handwritten digits.

Classification problem.



https://www.wolfram.com/mathematica/new-in-10/enhanced-image-processing/HTMLImages.en/handwritten-digits-classification/smallthumb_10.gif

MNIST Dataset



https://upload.wikimedia.org/wikipedia/commons/thumb/2/27/MnistExamples.png/320px-MnistExamples.png

Input

- 28x28 pixel images
- Training set
  - 60,000 samples
  - (60000, 28, 28)
- Testing set
  - 10,000 samples
  - (10000, 28, 28)

Target

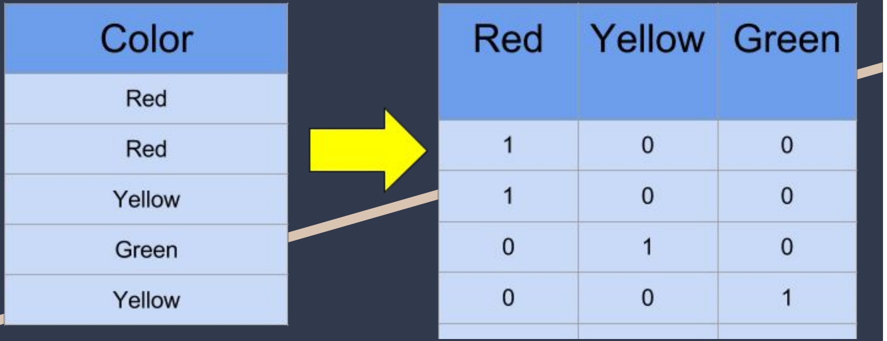
- Array of size 60,000/10,000
- 0-9 digits

Data Processing

Further process data in a way for easier training:

1. Flatten input for simplicity
   a. Reduce dimensionality
   b. (60000, 28, 28)
   c. (60000, 784)

2. Convert target to one-hot encoding for categorical data
   a. Non-ordinal
   b. [0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]
   c. 5



Color
Red
Red
Yellow
Green
Yellow

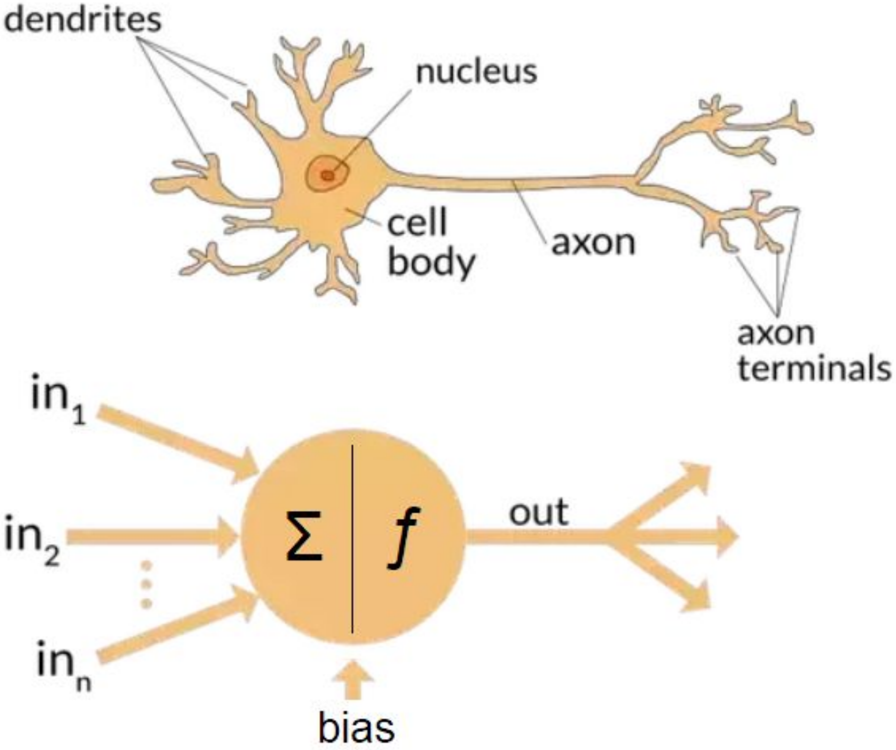
Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

https://www.kaggle.com/dansbecker/using-categorical-data-with-one-hot-encoding

Neurons: Analogy

The biological reference of how neural networks came to be.

How signals are transmitted.



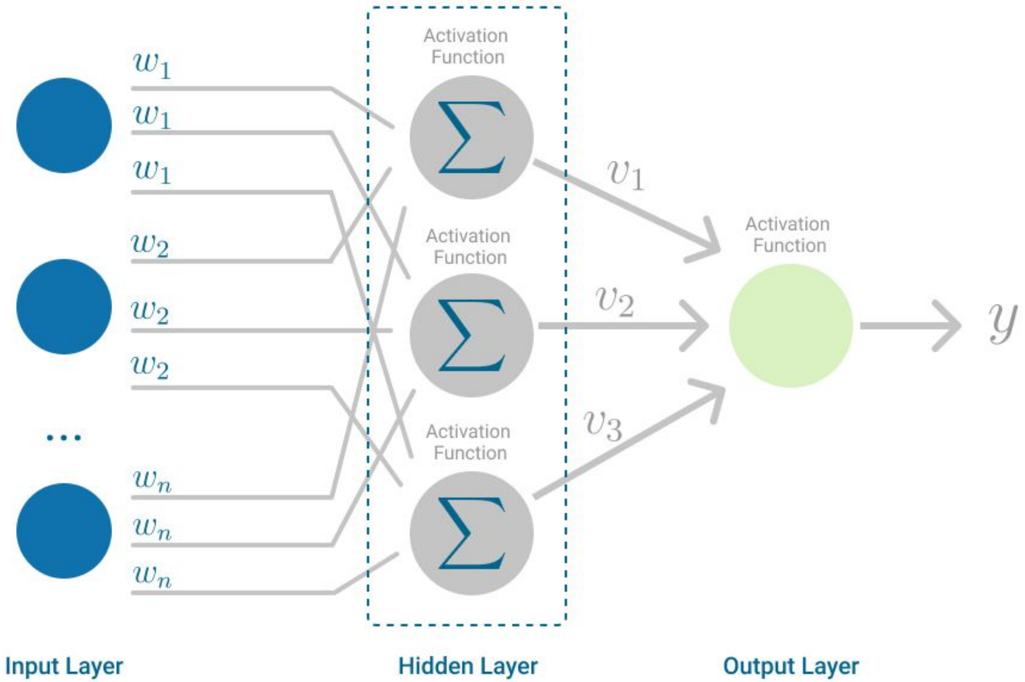
https://miro.medium.com/max/610/1*SJPacPhP4KDEB1AdhOFy_Q.png

Multi-Layered Perceptron (MLP)

Neuron: compute output based on inputs and weights

Weights: what will be optimized

Activation function: controls when neuron fires

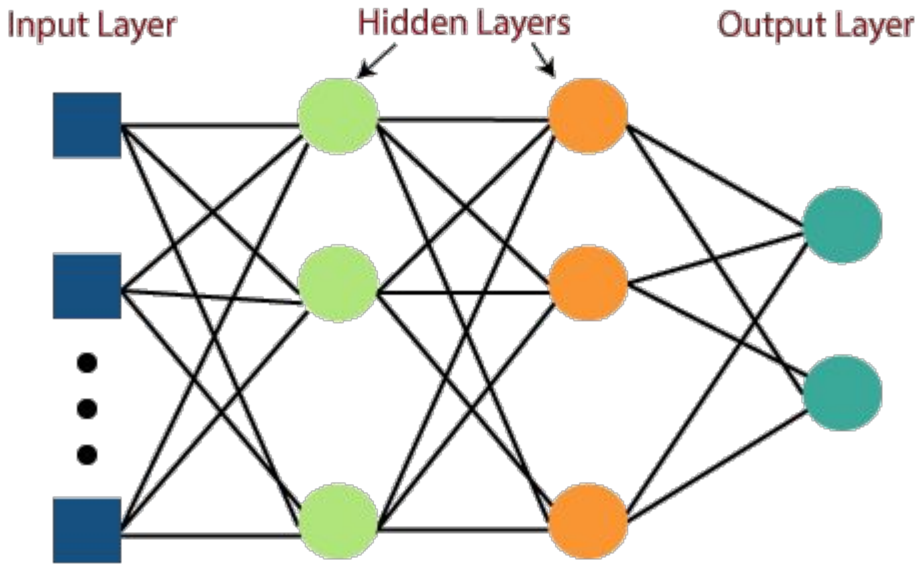


https://miro.medium.com/max/1400/1*MF1q2Q3fbpYIXX8fZUiwpA.png

MLP Structure

Layers

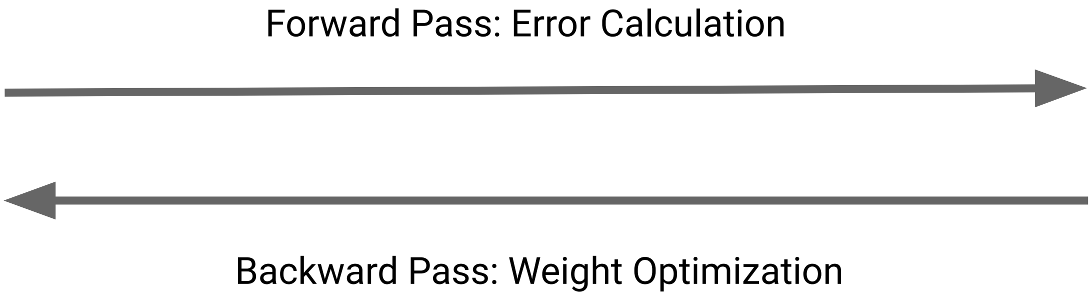
1. **Input**
   a. Initial layer to get input
2. **Hidden**
   a. Performs meaningful computations
3. **Output**
   a. Corresponding format



https://static.javatpoint.com/tutorial/tensorflow/images/multi-layer-perceptron-in-tensorflow.png

Model Training

- **Loss function**: calculate the error between prediction and label
- **Backpropagation**: send loss through network to adjust weight
- **Gradient descent**: optimize weights based on loss



Forward Pass: Error Calculation

Backward Pass: Weight Optimization

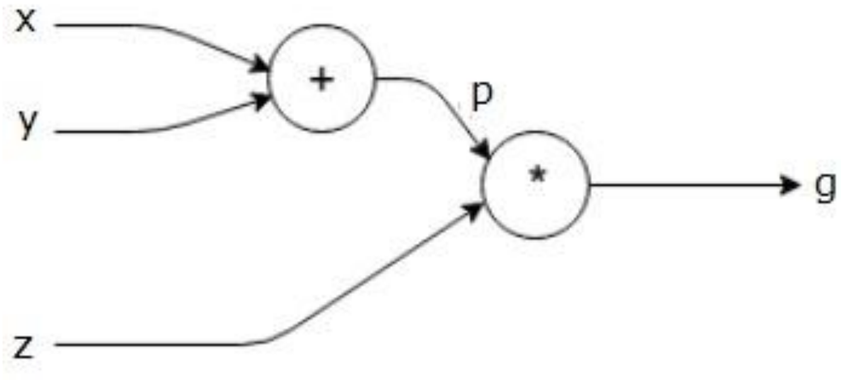
Tensorflow Introduction

Tensorflow: describe computational graphs

Graph: network of nodes

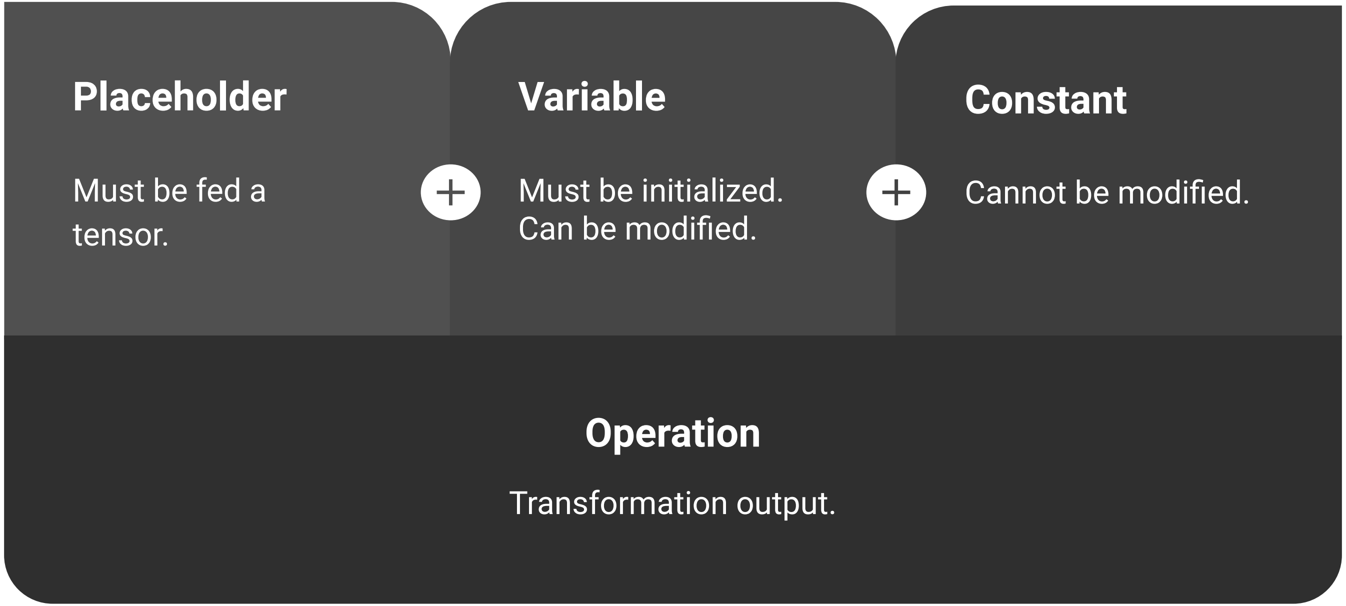
Node: operations with inputs and outputs

Tensors: multi-dimensional arrays



https://www.tutorialspoint.com/python_deep_learning/images/computational_graph_equation2.jpg

Input Types



Placeholder

Must be fed a tensor.

+

Variable

Must be initialized.
Can be modified.

+

Constant

Cannot be modified.

Operation

Transformation output.

Building Models

Placeholders	Features & Target
Probability	Softmax
Loss	Cross-Entropy Loss
Trainable Variable	Weight

Optimizer	tf.train.GradientDescentOptimizer()
	tf.train.AdamOptimizer()