Using Python and TensorFlow for Machine Learning

UNESP – 1st December 2022

Dr Ian Grout

Department of Electronic and Computer Engineering, University of Limerick, Limerick, V94 T9PX, Ireland

Email: Ian.Grout@ul.ie

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Seminar overview

Context	• Part 1
	  • Introduction.
	  • What is artificial intelligence?
	  • What is machine learning / deep learning?
		• Part 1
		  • 45 minutes presentation.
		  • 5 minute break.

Getting started	• Part 2
	  • TensorFlow and TensorFlow Lite (machine learning on edge devices).
	  • Installing Python and TensorFlow.
	  • Mathematical operations with NumPy and TensorFlow.
		• Part 2
		  • 45 minutes presentation.
		  • 5 minute break.

Image classification example	• Part 3
	  • Work through a classification example with TensorFlow, Keras, and the MNIST dataset.
		• Part 3
		  • 40 minutes presentation.
		  • 10 minutes Conclusions and Q&A.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

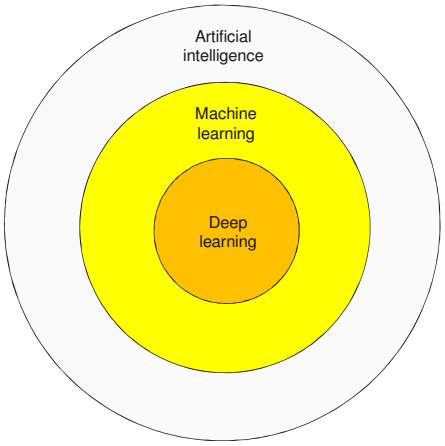
Part 1

- Introduction.
- What is artificial intelligence?
- What is machine learning / deep learning?

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Introduction

- Machine learning has gained significant interest in the last few years in a range of potential applications.
- Machine learning is based on specific principles implemented using suitable electronic hardware and software.
- Programming for machine learning differs from “traditional” programming.
- Machine learning deals with a large amount of data which needs to be evaluated using supervised or unsupervised learning approaches.



UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

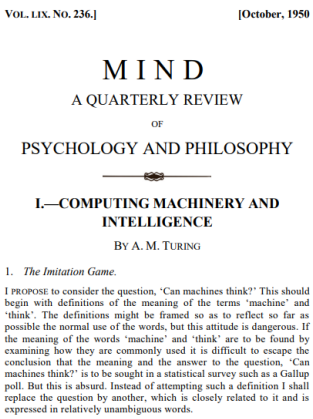
What is artificial intelligence

- A big question as what is intelligence?
- What is artificial intelligence (AI)?
- Early AI systems were referred to a “expert systems” and used a large number of rules listed by human programmers:
  - For example, refer to the Prolog language (https://www.swi-prolog.org/)
- Much more now than an “expert system”.
- More complicated problems use deep learning with neural networks.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

The Turing test

- The Turing test.
- Developed by Alan Turing in 1950 (University of Manchester, UK).
- A test of a machine's ability to exhibit intelligent behavior that is equivalent to, or indistinguishable from, that of a human.



VOL. LIX. NO. 236.] [October, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE

BY A. M. TURING

1. *The Imitation Game.*

I PROPOSE to consider the question, ‘Can machines think?’ This should begin with definitions of the meaning of the terms ‘machine’ and ‘think’. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words ‘machine’ and ‘think’ are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, ‘Can machines think?’ is to be sought in a statistical survey such as a Gallup poll. But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words.

Available:
https://academic.oup.com/mind/article/LIX/236/433/986238?login=true

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Artificial general intelligence

- A machine can perform the tasks that a human can.
- Far from a solved problem.
- Artificial general intelligence: AGI.

"*A machine with true AGI would be able to perform any intellectual task a human being can.*"

Ref: https://www.forbes.com/sites/forbestechcouncil/2018/06/18/artificial-general-intelligence-breakthroughs-to-watch-out-for-in-2018/#58e091166f83

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Concerns with AI and machine learning

- Ethics – what should intelligent machines do and not do?
- Data accuracy?
- Data bias – unintentional and intentional?
- Data security?
- Political and commercial advertisement targeting: influencing, bias?
- Just because something can be done, should it be done?
- Unintended consequences?
- Do we need to control who can develop AI solutions?

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

What is machine learning?

Process of teaching a computer system to make an accurate prediction when provided with data.

Machine learning (ML) is a subset of artificial intelligence (AI).

Machine learns to complete a certain task rather than being explicitly programmed to do so.

- Deals with a large amount of data. The input data might be biased, and/or might not be precise, complete or accurate.
- Input data can be in different formats – consider how people construct sentences, speak (language, dialect, accent) and write.
- Usually a need to pre-process the data into a suitable format and possibly reduce the amount of data required.
- Traditional computer software vs. machine learning (ML) software. ML software is not coded with solutions rather it is programmed with models of how to reliably create a solution based on provided data.
- Machine learning requires a large amount of reliable and unbiased data to determine a reliable solution.
- The data may be labelled (in supervised learning) or unlabeled (in unsupervised learning).

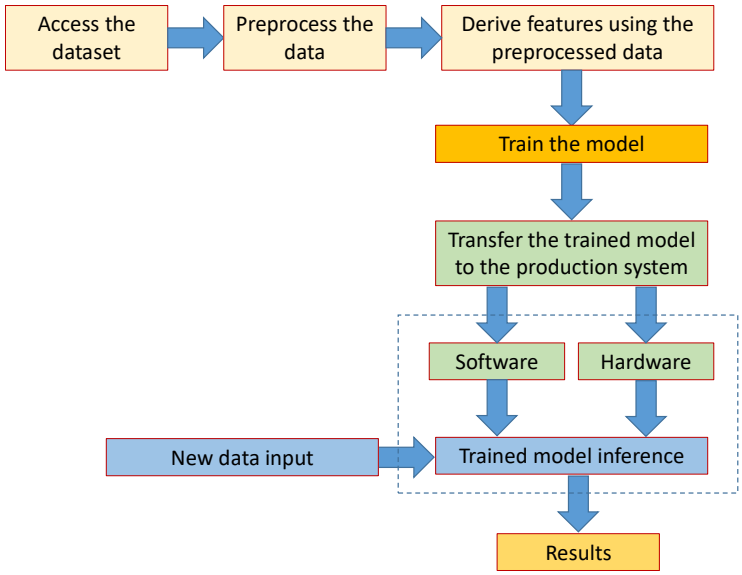
UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

What is deep learning?

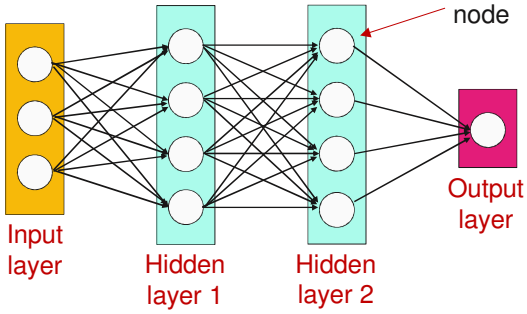
- A subset of machine learning in which tasks as separated and distributed onto machine learning algorithms organized in consecutive layers.
- Mimics the behavior of people.
- A computer model learns to perform classification tasks directly from images, text or sound.
- Aims to improve the accuracy of the output.
- Most deep learning methods use neural networks. Deep learning models are often referred to as deep neural networks (DNNs).

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Machine learning workflow



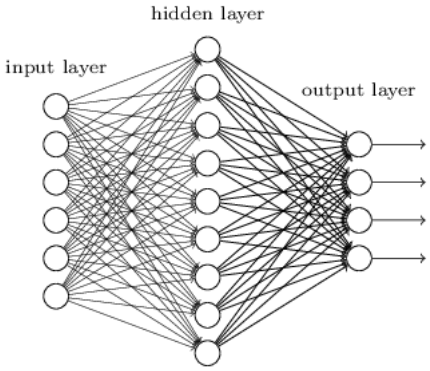
Neural networks



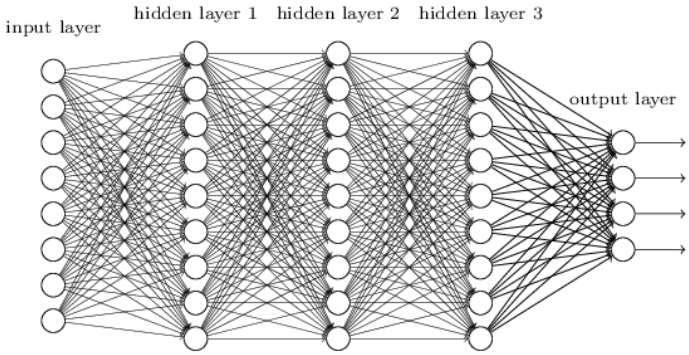
- Deep learning is based on neural networks.
- Structure of a neural network.
- Training (or fitting) will aim to identify suitable values of the parameters that can be trained within each of the layers.
- Note: There is a difference between a parameter and a hyperparameter.

- Neural networks consist of input and output layers, as well as in most cases, one or more hidden layers consisting of units that transform an input into something that the output layer can use.
- Note that not all machine learning requires the development and training of a neural network.

Neural networks



Artificial neural network



Deep neural network

Images from: http://neuralnetworksanddeeplearning.com/chap5.html

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Neural networks

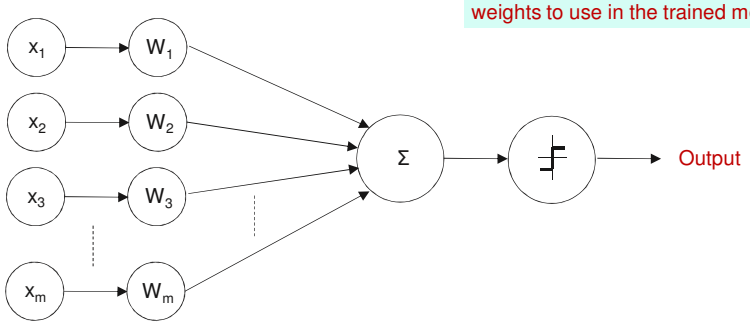
- ANN: Artificial neural network.
  - A system of (hardware and/or software) modelled after the operation of neurons in the human brain.
- DNN: Deep neural network.
  - A neural network with multiple internal layers suited for deep learning applications.
- CNN: Convolutional neural network.
  - A class of deep, feed-forward artificial neural networks and are most commonly used to analyze images.
- RNN: Recurrent neural network.
  - A class of artificial neural networks where connections made between nodes form a directed graph along a sequence.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Neural networks: Artificial neuron (node) - 1

- A node is a place where a computation occurs. It is loosely patterned on the human neuron. It combines input from the data with a set weights that assign significance. These input-weight products are summed and the sum is passed through an activation function.

Model training is aimed to take an algorithm with training data and identify optimal values for the weights to use in the trained model.



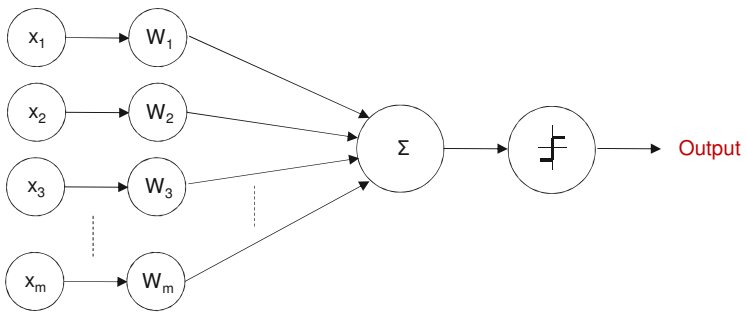
Inputs Weights Summation Activation function

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Neural networks: Artificial neuron (node) - 2

Different activation functions for different purposes

Multiply and add

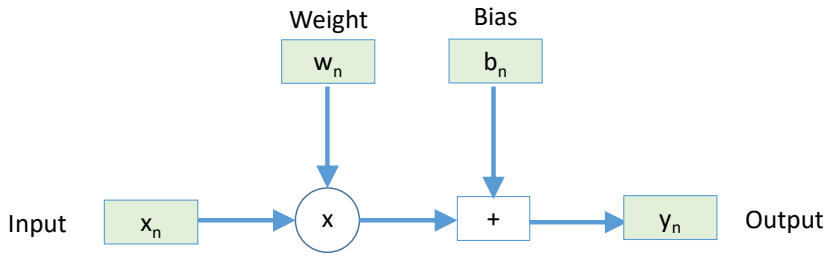


Inputs Weights Summation Activation function

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Neural networks: Artificial neuron (node) - 3

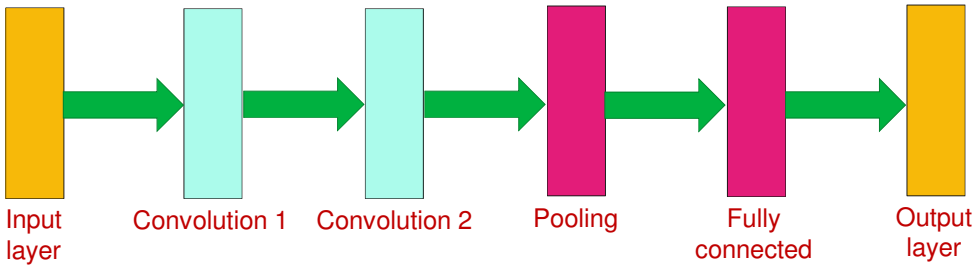
- Consider:
  - One input which is multiplied by a weight.
  - Add a constant bias.
  - No activation function.
- Question: *What is a perceptron?*



$$y_n = (w_n . x_n) + b_n$$

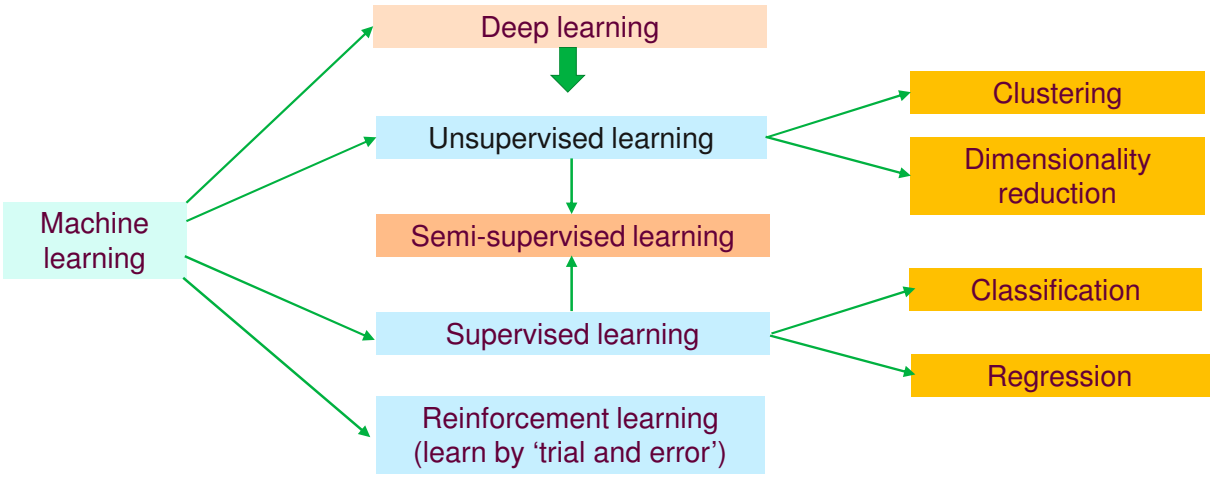
Example of a simple CNN architecture

- Different operations exist in each layer with a specific purpose.
- Initially, a suitable machine learning algorithm is defined and training data applied to the algorithm: the model to create is initially not trained and the trainable parameters in each layer are at default values.
- The process of training a machine learning model involves providing a machine learning algorithm with training data to learn from. The machine learning model is created by the training process.
- Model training will aim to identify optimal values for the trainable parameters within the model.
- Once trained, new data can be applied to the model and a prediction made (inference).



The CNN is a type of machine learning algorithm.

Categories of machine learning?

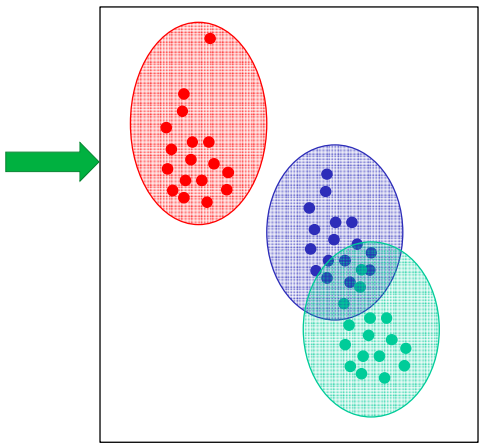


Useful online resource: https://docs.aws.amazon.com/machine-learning/latest/dg/training-ml-models.html

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Unsupervised learning

- Unsupervised learning finds patterns or structures in a provided dataset or datasets.
- It is used to draw inferences from datasets consisting of input data without any labels that can be used identify the data.
- Clustering is the most common unsupervised learning technique and is used for data analysis in order to find patterns or groupings in the provided data.
- Applications for clustering include gene sequence analysis, market research and object recognition.
- Dealing with overlapping groups of data and data outlier points?
- The number of input variables or features for a dataset is referred to as its dimensionality.
- Dimensionality reduction refers to techniques that are used to reduce the number of input variables in a dataset.



UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Supervised learning

- Supervised learning uses labelled data for training purposes. It is typical for a large amount of labelled data to be used.
- Both classification and regression form the basis of supervised learning where:
- Classification where the output is a category (or class) such as a color.
- Regression where the output is a real value such as cost or weight.
- Labelling of the data is a vast task. There are examples of labelled image datasets such as the Google Open Images Dataset (https://storage.googleapis.com/openimages/web/index.html)

Data value + Label

University of Limerick, Antony Gormley sculpture in the snow.

University of Limerick, Main Reception

Aerial view of the University of Limerick campus

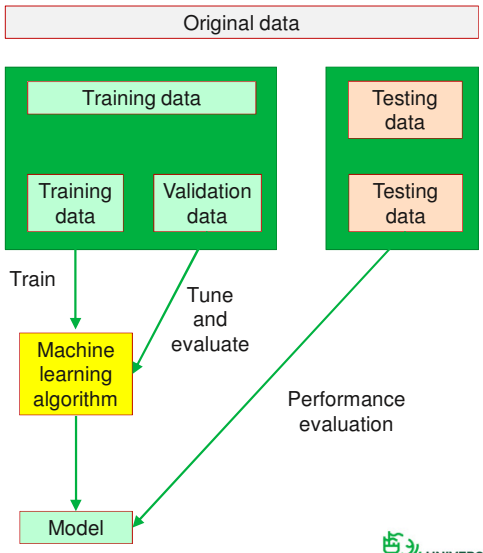
UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Supervised learning

- Labelled data will be used for three key tasks:
  1. Training. The sample of data used to fit the model.
  2. Validation. The sample of data used to provide an unbiased evaluation of a model fit on the training dataset.
  3. Testing. The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.

How much of the provided data should be used for training and for validation?

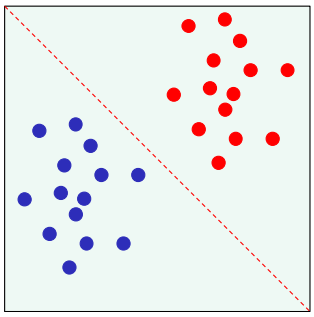
Are the words training and validation sometimes used to mean the same thing?



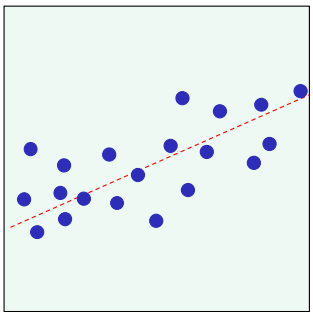
UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Supervised learning

Classification



Regression



$$y_n = (w_n . x_n) + b_n$$

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Model fitting = model training (1)

- Each machine learning algorithm has a basic set of parameters that can be changed to improve its accuracy. Before model training, initial values for the algorithm parameters are given. After model training, the parameters have been optimized.
- Model fitting is a measure of how well a machine learning model generalizes to similar data to that on which it was trained.
- During the fitting process, an algorithm is run using data for which the target variable is known (the "labelled" data) to produce a machine learning model:
    - An untrained model is the algorithm before training.
    - A trained model is the algorithm with optimized parameters after training.
- Underfitting:
    - The model cannot adequately capture the underlying structure of the data.
- Overfitting:
    - The model corresponds too closely, or exactly, to a particular set of data. It may fail to fit additional data or predict future observations reliably.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Model fitting = model training (2)

- Number of epochs:
  - An epoch is the one complete pass of the entire training dataset by the machine learning algorithm.
- Batch size:
  - A training dataset can be divided into one or more batches.
  - The batch size is the number of training examples used in a single iteration of the machine learning algorithm.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Data requirements

- Amount of data?
- How biased or unbiased is the data?
- How accurate is the data?
- Who collects the data?
- Who processes the data?
- Who can use the data?
- Confidentiality, privacy, copyright issues?

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Part 2

- Mathematical operations with NumPy and TensorFlow.
- TensorFlow and TensorFlow Lite (machine learning on edge devices).
- Installing Python and TensorFlow.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Programming languages for machine learning

- Programming languages used in machine learning?
- These include:
  - Python.
  - Java.
  - R.
  - C.
  - C++.
  - JavaScript.
  - Scala.
  - Julia.
- Pros and cons of each language (support, speed, compiled vs interpreted code)?

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Programming languages for machine learning

- Why use Python for machine learning? Reasons include …
  - Wide support and community involvement: popular language for engineering and science.
  - Less code to write when using prebuilt libraries/modules/functions.
  - Examples to learn from.
  - Operating system/platform independence.
  - Flexibility.
- Modules (packages or libraries) include:
  - NumPy.
  - SciPy.
  - SciKit-Learn.
  - TensorFlow.
  - Keras.
  - Theano.
  - PyTorch.

Python references:

- https://www.python.org/
- https://www.w3schools.com/python/

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

What is TensorFlow

- TensorFlow is an open source software library created by Google. It was developed for numerical computation using data flow graphs.
  - https://www.tensorflow.org/
- TensorFlow has APIs available in several languages both for constructing and executing a TensorFlow graph.
- TensorFlow can be used with Python.
- Keras can be used with Python and also is integrated into TensorFlow.
- Now version 2 which is a significant change from version 1.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

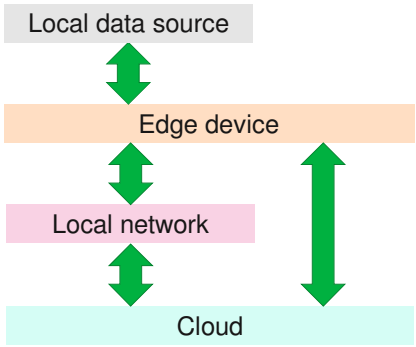
TensorFlow v1 to TensorFlow v2

- TensorFlow version 1 to TensorFlow version 2:
  - Integration of Keras (https://keras.io/ and https://www.tensorflow.org/api_docs/python/tf/keras)
  - No sessions … more Python-*like* programming.
  - Eager execution enabled by default.
  - TensorFlow v1 code can be migrated to v2.
  - TensorFlow v1 code can be run in v2 using `tf.compat.v1`
  - YouTube video https://www.youtube.com/watch?v=JmSNUeBG-PQ

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

TensorFlow Lite

- An extension to TensorFlow for deploying machine learning models on mobile and edge devices:
  - https://www.tensorflow.org/lite
- Edge devices:
  - Edge devices are computing devices that process local level data and transmit data to the local network and the cloud. Edge devices can be microcontrollers, microprocessors, digital signal processors, graphics processing units (GPUs), field programmable gate arrays (FPGAs), application specific integrated circuits (ASICs), other computing platforms, custom hardware and software solutions, …
- Can be deployed on Android and iOS operating systems:
  - https://www.tensorflow.org/lite/android
  - https://www.tensorflow.org/lite/guide/ios
- Can be deployed on microcontrollers, for example the STM32:
  - https://www.tensorflow.org/lite/microcontrollers

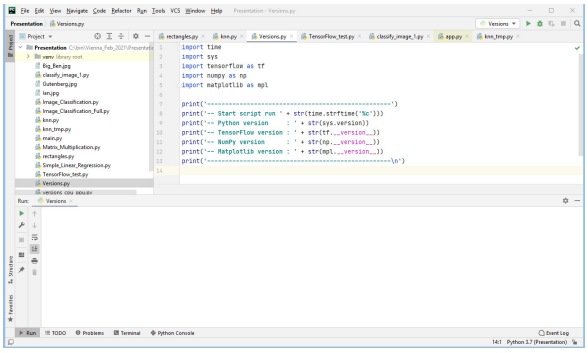


UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Setting-up and starting with TensorFlow

- Consider the scenario using:

1. PyCharm Community Edition 2022.2.3.
2. Creating PyCharm projects for code development using virtualenv virtual environment.
3. Python 3.8.
4. TensorFlow 2.11.0.
5. NumPy 1.23.5 [NumPy installed when installing TensorFlow].
6. Matplotlib 3.6.2.
7. Windows 64-bit 10 operating system with one CPU and no GPU.
8. Tensorboard: 2.11.0.



Other IDEs include Spyder, Google CoLab, Visual Studio Code, and Jupyter Notebook.

Python and TensorFlow versions:
https://www.tensorflow.org/install/pip

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Software versions can be important …

- Python script to identify the version of Python and the modules/packages used.
- Can be used to check the modules/packages have been installed OK.
- Installation of modules/packages in Python is performed using `pip` (a package manager for Python packages).

```
import time
import sys
import tensorflow as tf
import numpy as np
import matplotlib as mpl


def main():

    print('Test script to identify Python and package/module versions')

    print('--------------------------------------------------------')
    print('-- Start script run ' + str(time.strftime('%c')))
    print('-- Python version      : ' + str(sys.version))
    print('-- TensorFlow version  : ' + str(tf.__version__))
    print('-- NumPy version       : ' + str(np.__version__))
    print('-- Matplotlib version  : ' + str(mpl.__version__))
    print('--------------------------------------------------------\n')


if __name__ == "__main__":

    main()
```

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

NumPy and Matplotlib

- What is NumPy?
  - NumPy -> Numerical Python.
  - NumPy is a package for scientific computing with Python.
  - https://numpy.org/
- What is Matplotlib?
  - Matplotlib is a library for creating static, animated, and interactive visualizations in Python.
  - https://matplotlib.org/

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

NumPy example 1

- Scalars and arrays.

```
import numpy as np


def print_vals(data):

    if np.isscalar(data):
        print('-- SCALAR')
        print(type(data))
        print(data)

    if isinstance(data, np.ndarray):
        print('-- NDARRAY')
        print(type(data))
        print(data)
        print(data.shape)
        print(data.ndim)


A = 5
B = 10.0
C = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print('-- A ----------------------')
print_vals(A)
print('-- B ----------------------')
print_vals(B)
print('-- C ----------------------')
print_vals(C)
print('------------------------')
```

```
-- A ----------------------
-- SCALAR
<class 'int'>
5
-- B ----------------------
-- SCALAR
<class 'float'>
10.0
-- C ----------------------
-- NDARRAY
<class 'numpy.ndarray'>
[[1 2 3]
 [4 5 6]
 [7 8 9]]
(3, 3)
2
------------------------
```

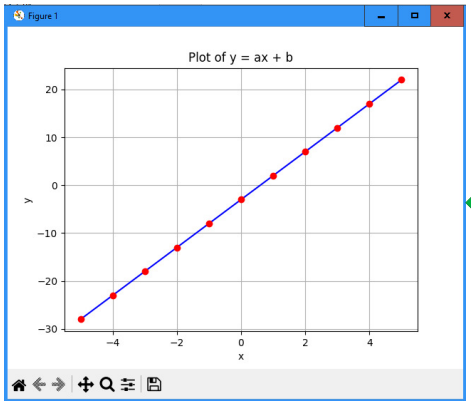
UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

NumPy example 2

- Consider the equation:

  `y = ax + b`

- Where:
  - a = 5.0
  - b = -3.0
  - x varies from -5.0 to +5.0 in steps of 1.0




```
import numpy as np
import matplotlib.pyplot as plt

a = 5.0
b = -3.0
x = np.linspace(-5.0, 5.0, 11, endpoint=True)

# y = (a*x) + b
y = np.add(np.multiply(a, x), b)

print(a)
print(b)
print(x)
print(y)

plt.plot(x, y, 'b-')
plt.plot(x, y, 'ro')
plt.grid(True)
plt.xlabel('x')
plt.ylabel('y')
plt.title('Plot of y = ax + b')
plt.show()
```

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Tensors

- TensorFlow is a framework to define and run computations involving tensors.
- A tensor is a generalization of vectors and matrices which can involve multiple dimensions.
- The rank of an object is its number of dimensions:

Rank	Entity	Details
0	Scalar	magnitude only
1	Vector	magnitude and direction
2	Matrix	m x n array
3	3-dimension cuboid	m x n x p array
n	n-dimensions	

- Introduction to Tensors https://www.tensorflow.org/guide/tensor

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Tensors

- A tensor is the mathematical representation of a physical quantity that can be characterized by its magnitude and multiple directions.
- Tensors are multi-dimensional arrays that, like vectors and matrices, can be used in linear algebra.
- A tensor is a generalization of scalars, vectors, and matrices.
- Tensors have a rank.
- Tensor algebra allows mathematical operations on arrays of any dimension.
- In this work, tensors are considered as multi-dimensional arrays.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

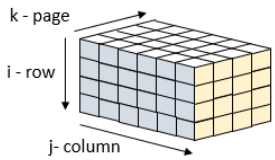
Tensors

- Consider tensors where the elements are scalar quantities.
- Describing a tensor by following the same notation as matrix notation
- A capital letter represents the tensor.
- A lower case letter with subscript integers represent scalar quantities within the tensor.
- Consider a rank 3 tensor representing a cuboid using matrix notation.
- *Now consider space and time … 4 dimensions.*

Tensor **A**

Element within **A**:

a_{ijk}



UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

NumPy: Tensor example

- Consider a matrix.
- The matrix is a *m x n* array and a rank 2 tensor.
- Consider this *3 x 3* matrix in Python and NumPy, creating a NumPy array..

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
import numpy as np

A = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print(A)
print(A.shape)
print(A.ndim)
```

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
(3, 3)
2
```


TensorFlow example 1

```
A = tf.constant(5)
B = tf.constant(6.0)
C = tf.constant([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
D = tf.constant([[1, 2, 3], [4, 5, 6], [7, 8, 9]], dtype = tf.dtypes.float32)

print(A)
print(B)
print(C)
print(D)
```

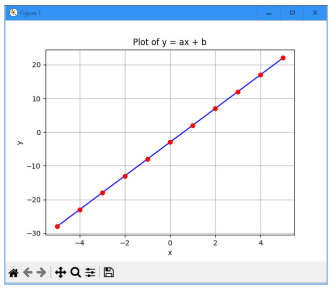
$$A = 5 \qquad C = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$$B = 6 \qquad D = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
tf.Tensor(5, shape=(), dtype=int32)
tf.Tensor(6.0, shape=(), dtype=float32)
tf.Tensor(
[[1 2 3]
 [4 5 6]
 [7 8 9]], shape=(3, 3), dtype=int32)
tf.Tensor(
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]], shape=(3, 3), dtype=float32)
```

TensorFlow example 2

- Consider the equation:
  - y = ax + b
- Where:
  - a = 5.0
  - b = -3.0
  - x varies from -5.0 to +5.0 in steps of 1.0




```
import tensorflow as tf
import matplotlib.pyplot as plt

a = tf.constant(5.)
b = tf.constant(-3.)
x = tf.constant(tf.linspace(-5.0, 5.0, 11))

# y = (a*x) + b
y = tf.add(tf.multiply(a, x), b)

print(a)
print(b)
print(x)
print(y)

plt.plot(x, y, 'b-')
plt.plot(x, y, 'ro')
plt.grid(True)
plt.xlabel('x')
plt.ylabel('y')
plt.title('Plot of y = ax + b')
plt.show()
```

```
tf.Tensor(5.0, shape=(), dtype=float32)
tf.Tensor(-3.0, shape=(), dtype=float32)
tf.Tensor([-5. -4. -3. -2. -1.  0.  1.  2.  3.  4.  5.], shape=(11,), dtype=float32)
tf.Tensor([-28. -23. -18. -13.  -8.  -3.   2.   7.  12.  17.  22.], shape=(11,), dtype=float32)
```

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

TensorFlow: Eager execution

- "TensorFlow's eager execution is an imperative programming environment that evaluates operations immediately, without building graphs: operations return concrete values instead of constructing a computational graph to run later."
  - REF: https://www.tensorflow.org/guide/eager
- In TensorFlow v1, the concept of sessions was introduced.
- In TensorFlow 2, sessions have been removed and Eager execution enabled by default.
  - tf.executing_eagerly()

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

TensorFlow GPU support

```
import tensorflow as tf
from tensorflow.python.client import device_lib


def main():

    print(device_lib.list_local_devices())

    local_devices = tf.config.experimental.list_physical_devices()

    print('Num CPUs Available: ',
          len(tf.config.experimental.list_physical_devices('CPU')))

    for x in local_devices:
        if x.device_type == 'CPU':
            print(x)

    print('Num GPUs Available: ',
          len(tf.config.experimental.list_physical_devices('GPU')))

    for x in local_devices:
        if x.device_type == 'GPU':
            print(x)


if __name__ == "__main__":

    main()
```

- TensorFlow operations can be assigned to different connected processors:
  - CPU.
  - GPU.
  - TPU (Tensor Processing Unit).
- Consider the code to the left that identifies connected CPUs and GPUs.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

TensorFlow: tensors, variables and constants

- Tensors
  - tf.Tensor
  - https://www.tensorflow.org/api_docs/python/tf/Tensor
  - A tensor is a multidimensional array of elements represented by a tf.Tensor object.
- Variables
  - tf.Variable
  - https://www.tensorflow.org/guide/variable
  - TensorFlow variables are created and tracked via the tf.Variable class.
  - A TensorFlow variable is the recommended way to represent shared, persistent states the program manipulates.
- The most important difference between Variables and Tensors is mutability. Unlike tensors, values in a Variable object can be updated (e.g., with the assign() function). The values of tensor objects cannot be updated It is only possible to create a new Tensor object with the new values.
- Constants
  - tf.constant
  - https://www.tensorflow.org/api_docs/python/tf/constant
  - Creates a constant tensor from a tensor-like object.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

TensorFlow: tensor rank, size, and shape

- tf.rank
  - Returns the rank of a tensor.
    - https://www.tensorflow.org/api_docs/python/tf/rank?hl=bn
- tf.size
  - Returns the size of a tensor.
    - https://www.tensorflow.org/api_docs/python/tf/size?hl=bn
- tf.shape
  - Returns a tensor containing the shape of the input tensor.
    - https://www.tensorflow.org/api_docs/python/tf/shape?hl=bn

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

NumPy compatibility

- A Tensor is a multi-dimensional array.
- Tensors are similar to NumPy ndarray objects: tf.Tensor objects have a data type and a shape.
- Differences between NumPy arrays and tf.Tensors include:
  - Tensors computations can be supported with the GPUs and TPUs.
  - Tensors are immutable.
- NumPy compatibility: can convert between a TensorFlow tf.Tensors and a NumPy ndarray is easy:
  - TensorFlow operations automatically convert NumPy ndarrays to Tensors.
  - NumPy operations automatically convert Tensors to NumPy ndarrays.
  - Tensors are explicitly converted to NumPy ndarrays using their .numpy() method.
- tf.convert_to_tensor
  - Converts given value to a Tensor.
  - https://www.tensorflow.org/api_docs/python/tf/convert_to_tensor

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

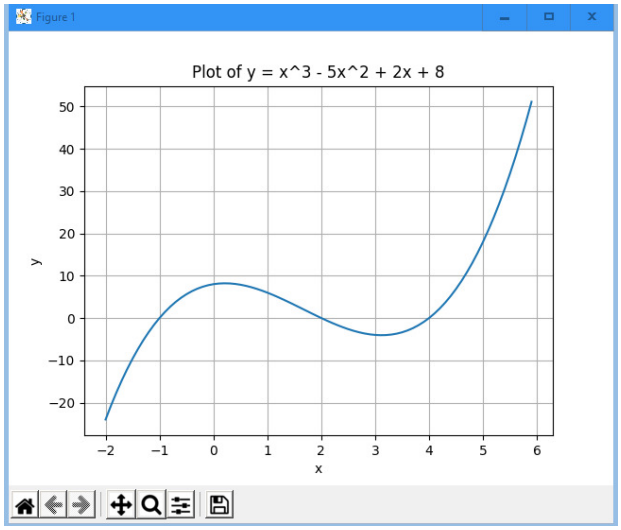
Example 1: Solving an equation in TensorFlow

- Consider the equation:

$$y = x^3 - 5x^2 + 2x + 8$$

```
x = tf.range(-2., 6., 0.1, dtype=tf.float32)
```

```
y2 = tf.math.add(tf.math.add(tf.math.subtract(
    tf.pow(x, 3),
    tf.math.multiply(5., tf.math.square(x))),
    tf.math.multiply(2., x)),
    tf.constant(8.))
```



UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

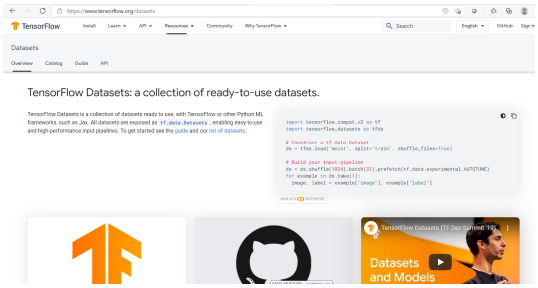
Part 3

- Work through a classification example with TensorFlow, Keras, and the MNIST dataset.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Image recognition datasets

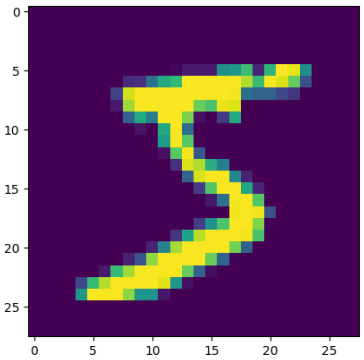
- TensorFlow can use available datasets.
- TensorFlow Datasets: a collection of ready-to-use datasets.
- https://www.tensorflow.org/datasets
- For example, this example uses the Imagenet Inception image recognition model:
- http://download.tensorflow.org/models/image/imagenet/inception-2015-12-05.tgz
- Refer also to Keras documentation:
- https://www.tensorflow.org/api_docs/python/tf/keras/applications/InceptionV3



The Inception v3 model is a deep convolutional neural network that has been pre-trained for the ImageNet Large Visual Recognition Challenge (http://www.image-net.org/challenges/LSVRC/).

MNIST dataset

- Widely used as an introduction to classification (supervised learning).
- It is a dataset consisting of 60,000 28x28 grayscale images of the 10 digits (0 to 9), along with a test set of 10,000 images.
- Ref:
  - https://keras.io/api/datasets/mnist/
  - https://yann.lecun.com/exdb/mnist/ (MNIST homepage – password protected).
  - https://www.tensorflow.org/datasets/keras_example
  - https://www.tensorflow.org/datasets/catalog/mnist
- Note, when the dataset is loaded and the dataset has not been previously installed on the local computer, the dataset will be downloaded and installed.

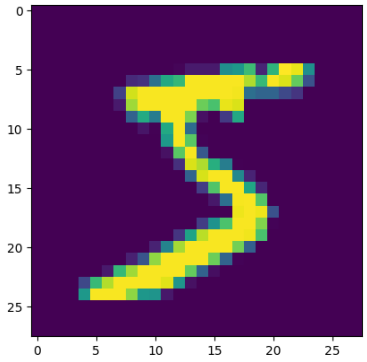


1st image in MNIST dataset viewed using Matplotlib.

This is a handwritten number 5.

Example Python code to read and plot an image

```
import tensorflow as tf
import matplotlib.pyplot as plt

dataset = tf.keras.datasets.mnist

x_train, y_train), (x_test, y_test) = dataset.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

print('-- Train and Test ----------------------------')
print('>> Train: x=%s, y=%s' % (x_train.shape, y_train.shape))
print('>> Test : x=%s, y=%s' % (x_test.shape, y_test.shape))

print('-- x_train[0] ---------------------------------')
print(x_train[0])
print('-- y_train[0] ---------------------------------')
print(y_train[0])
print('-----------------------------------------------')

plt.imshow(x_train[0])
plt.show()
```



1st image in MNIST dataset viewed using Matplotlib.

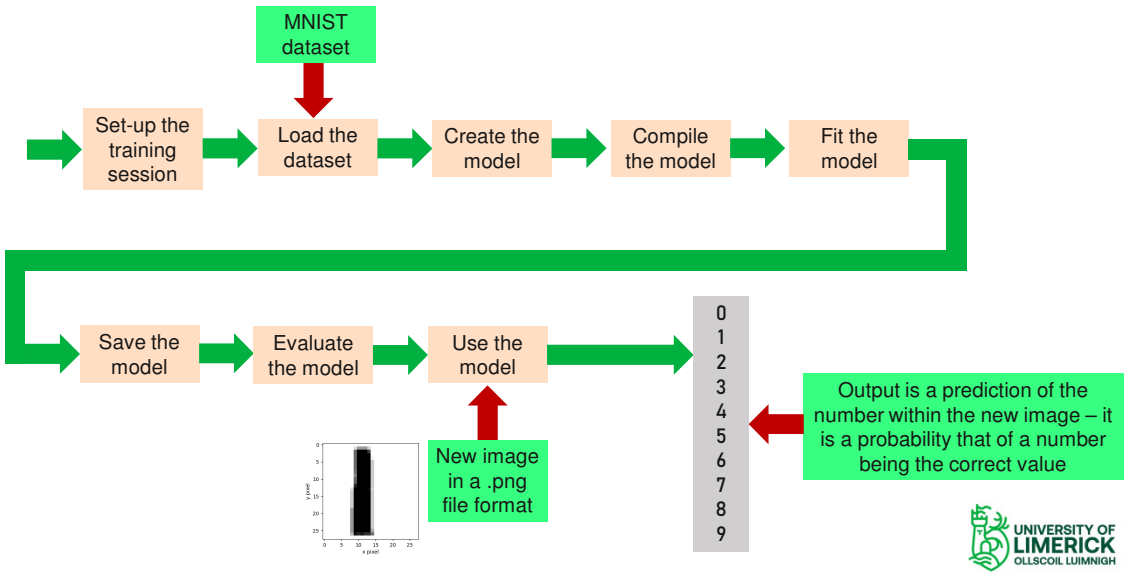
This is a handwritten number 5.

```
-- Train and Test ----------------------------
>> Train: x=(60000, 28, 28), y=(60000,)
>> Test : x=(10000, 28, 28), y=(10000,)
```

```
-- y_train[0] ---------------------------------
5
```

UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Creating, training and using a model based on a CNN



UNIVERSITY OF LIMERICK OLLSCOIL LUIMNIGH

Set-up the training session

```
# ------------------------------------------------------
# Modules to import
# ------------------------------------------------------

import tensorflow as tf

import matplotlib.pyplot as plt

import cv2
```

OpenCV – Install using
`pip install opencv-python`

```
# ------------------------------------------------------
# Set-up
# ------------------------------------------------------

dataset = tf.keras.datasets.mnist

model_dir = 'model_directory'

no_of_epochs = 10
```

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Load the dataset

```
# ------------------------------------------------------
# Load the dataset
# ------------------------------------------------------

(x_train, y_train), (x_test, y_test) = dataset.load_data()

x_train, x_test = x_train / 255.0, x_test / 255.0

print('>> Train: x=%s, y=%s' % (x_train.shape, y_train.shape))
print('>> Test : x=%s, y=%s' % (x_test.shape, y_test.shape))
```

https://www.tensorflow.org/api_docs/python/tf/keras/datasets/mnist/load_data

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Create the model

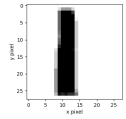
```
# ---------------------------------------------------
# Create the model
# ---------------------------------------------------

model = tf.keras.models.Sequential(

    [tf.keras.layers.Flatten(input_shape=(28, 28)),

     tf.keras.layers.Dense(units=512, activation='relu'),

     tf.keras.layers.Dropout(0.2),

     tf.keras.layers.Dense(10, activation='softmax')]

)
```

https://www.tensorflow.org/api_docs/python/tf/keras/Sequential

A simple CNN.

What would be the best structure for the CNN?

What would the selection criteria be?



28 x 28
grayscale image

Flatten the array → Dense layer → Dropout layer → Dense layer

0
1
2
3
4
5
6
7
8
9

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

A better model?

```
from tensorflow import keras
from tensorflow.keras import layers
```

```
model = keras.Sequential(
    [
        keras.Input(shape=(28, 28, 1)),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dropout(0.5),
        layers.Dense(10, activation="softmax"),
    ]
)
```

Modified from: https://keras.io/examples/vision/mnist_convnet/

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Compile the model

```
# -------------------------------------------------
# Compile the model
# -------------------------------------------------

model.compile(

    optimizer='adam',
    loss='sparse_categorical_crossentropy',
    metrics=['accuracy']

)
```

Keras Optimizers – “adam”
https://keras.io/api/optimizers/

Keras Losses – “sparse_categorical_crossentropy”
https://keras.io/api/losses/

Keras Metrics – “accuracy”
https://keras.io/api/metrics/

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Fit the model

```
# -------------------------------------------------
# Fit the model
# -------------------------------------------------

history_callback = model.fit(
    x=x_train,
    y=y_train,
    epochs=no_of_epochs,
    validation_data=(x_test, y_test),
)

loss_history = history_callback.history['loss']
accuracy_history = history_callback.history['accuracy']
val_loss_history = history_callback.history['val_loss']
val_accuracy_history = history_callback.history['val_accuracy']

weights = model.get_weights()

print(model.summary())
print(weights)
print('loss         = ', loss_history)
print('accuracy     = ', accuracy_history)
print('val_loss     = ', val_loss_history)
print('val_accuracy = ', val_accuracy_history)
```

Validation here is performed using the test data.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Save the model

```
# ------------------------------------------------
# Save the model
# ------------------------------------------------

tf.keras.models.save_model(

    model,
    model_dir,
    overwrite=True,
    include_optimizer=True,
    save_format=None,
    signatures=None,

)
```

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

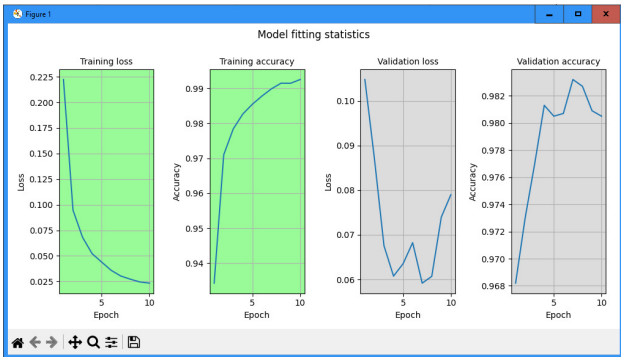
Evaluate the model

```
# ------------------------------------------------
# Evaluate the model
# ------------------------------------------------

test_loss, test_acc = model.evaluate(x_test, y_test, verbose=2)

print('Test Loss:     ', test_loss)
print('Test Accuracy: ', test_acc)

image_to_predict1 = cv2.imread('my_number_ex1.png')
image_to_predict1_gray = cv2.cvtColor(image_to_predict1, cv2.COLOR_BGR2GRAY)
gray_not = cv2.bitwise_not(image_to_predict1_gray)
image_to_predict1_28x28 = cv2.resize(gray_not, (28, 28), interpolation=cv2.INTER_AREA) / 255.0

print('-- Image to predict')
print('Shape of image ', image_to_predict1_28x28.shape)

probability_model = tf.keras.Sequential([model])
predictions = probability_model.predict(image_to_predict1_28x28.reshape(1, 28, 28))

print(predictions)
print('-- Predicted image')
print('Most probable image ', tf.argmax(predictions[0]))
```

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

```
# ------------------------------------------------
# Plot training results
# ------------------------------------------------
loss_history = history_callback.history['loss']
accuracy_history = history_callback.history['accuracy']
val_loss_history = history_callback.history['val_loss']
val_accuracy_history = history_callback.history['val_accuracy']

epochs = tf.linspace(1, no_of_epochs, no_of_epochs)

fig, axs = plt.subplots(1, 4, figsize=(10, 5))

axs[0].plot(epochs, loss_history)
axs[0].set_title('Training loss', fontsize=10)
axs[0].set_xlabel('Epoch', fontsize=10)
axs[0].set_ylabel('Loss', fontsize=10)
axs[0].grid(True)
axs[0].set_facecolor('palegreen')

axs[1].plot(epochs, accuracy_history)
axs[1].set_title('Training accuracy', fontsize=10)
axs[1].set_xlabel('Epoch', fontsize=10)
axs[1].set_ylabel('Accuracy', fontsize=10)
axs[1].grid(True)
axs[1].set_facecolor('palegreen')

axs[2].plot(epochs, val_loss_history)
axs[2].set_title('Validation loss', fontsize=10)
axs[2].set_xlabel('Epoch', fontsize=10)
axs[2].set_ylabel('Loss', fontsize=10)
axs[2].grid(True)
axs[2].set_facecolor('gainsboro')

axs[3].plot(epochs, val_accuracy_history)
axs[3].set_title('Validation accuracy', fontsize=10)
axs[3].set_xlabel('Epoch', fontsize=10)
axs[3].set_ylabel('Accuracy', fontsize=10)
axs[3].grid(True)
axs[3].set_facecolor('gainsboro')

plt.suptitle('Model fitting statistics')
plt.tight_layout()
plt.subplots_adjust(top=0.85)

plt.show()
```

Plot training and validation results (loss and accuracy)



UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Other useful actions

1. Load an already saved (trained) model.
2. Improve the trained model performance.
3. Save the model in different file formats.
4. Export the model for TensorFlow Lite. Implement trained model on an edge device.
5. Create documentation and graphics.

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Conclusions

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

Thank you

UNIVERSITY OF LIMERICK
OLLSCOIL LUIMNIGH

University of Limerick,
Limerick, V94 T9PX,
Ireland.
Ollscoil Luimnigh,
Luimneach,
V94 T9PX, Éire.
+353 (0) 61 202020

ul.ie