Introduction to ANN, TensorFlow and Keras

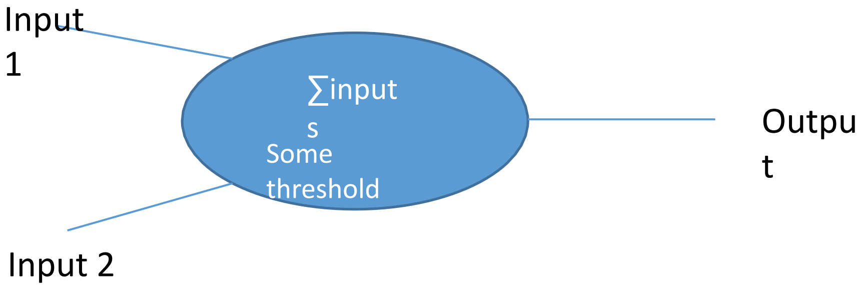
Topics Covered in the week

- Introduction to Artificial Neurons
- McCulloch – Pitts Neuron
- Rosenblatt’s Neuron
- Introduction of Artificial Neural Networks
- Introduction to TensorFlow and Keras
- Case Study

Introduction to Artificial Neurons

- The inception of research around the artificial neurons was inspired from the biological neuron.
- The objective was to develop a system that can do tasks on its own.
- Researchers tried to implement logic gates through the artificial neurons in early research.
- Important work was done by Warren McCulloch and Walter Pitts in this field. They created the McCulloch-Pitts neuron.

McCulloch-Pitts Neuron



- The McCulloch-Pitts neuron had binary inputs and outputs where inputs could be many but the output was just one.
- The input was multiplied with weights and summed up. That result would later be fed to a threshold function/step function to determine the output.

Rosenblatt’s Neuron

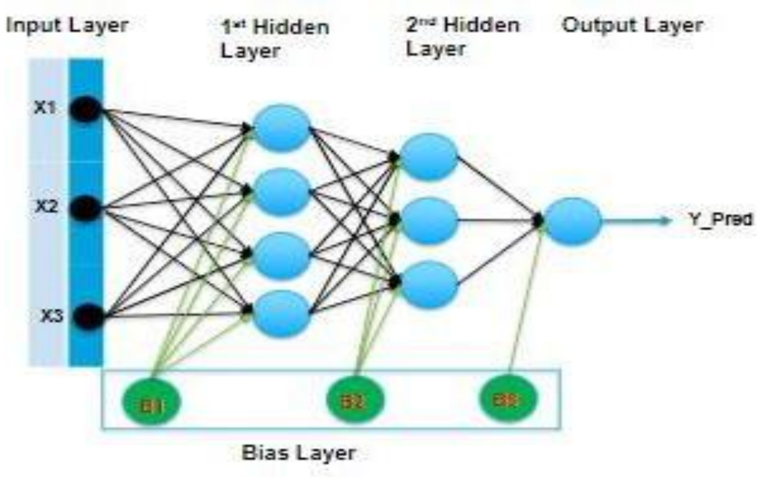
- Rosenblatt’s Neurons were an upgrade to the earlier McCulloch-Pitts neuron.
- Introduced different weights for inputs, bias input, captured errors and a learning rule.
- The neuron successfully captured the AND and OR gate w/o the need of manually setting the threshold.
- Thus the concept of learning from data was introduced.

However,

- The neuron could not capture the XOR gate, and gates where the targets were not linearly separable in the binary input(2-D) space.
- This led to further research and eventually the development of the ANN.

The Artificial Neural Networks

- It was demonstrated that a single neuron could not capture gates like XOR but a network of gates could do so.
- This gave birth to the ANN.



TensorFlow and Keras

- TensorFlow is an end-to-end open source machine learning library by Google that will help you develop and train ML and DL models.
- It offers easy model building support and makes development easy.
- Keras is a high-level neural networks API, written in Python.
- Tensorflow.Keras is the TensorFlow implementation of the Keras API specification.

Relevant Tensorflow Functions

- **Tensors** - A tensor is a generalization of vectors and matrices to potentially higher dimensions. Internally, TensorFlow represents tensors as n-dimensional arrays of base data types. Quoting from the official TensorFlow website, a tf.Tensor has the below properties:
  - A data type
  - A shape
    - *Each element in the Tensor has the same data type, and the data type is always known. The shape (that is, the number of dimensions it has and the size of each dimension) might be only partially known.*

TensorFlow and Keras - Contd

Special tensors

- Tf.Variable
- Tf.constant

Tensorflow allows the handling of mathematical operations quite similar to NumPy.

Some important functions –

- Tf.random.uniform – outputs random values from a uniform distribution
- Tf.random.normal – outputs random values from a normal distribution

Similarly, there are other random generators.

TensorFlow and Keras - Contd

Now we list some relevant operators. Note that there are a number of these and they can’t all be covered here. Just mentioning a few -

For multiplication on matrices:

- Tf.matmul – for matrix multiplication
- Tf.multiply – for element wise multiplication
- Tf.GradientTape – for finding the derivative of functions
- Tf.assign – assign values to a variable tensor
- Tf.assign_sub/ Tf.assign_add – sub/add to the given variable tensor.
- Tf.math – for other math operations

Keras

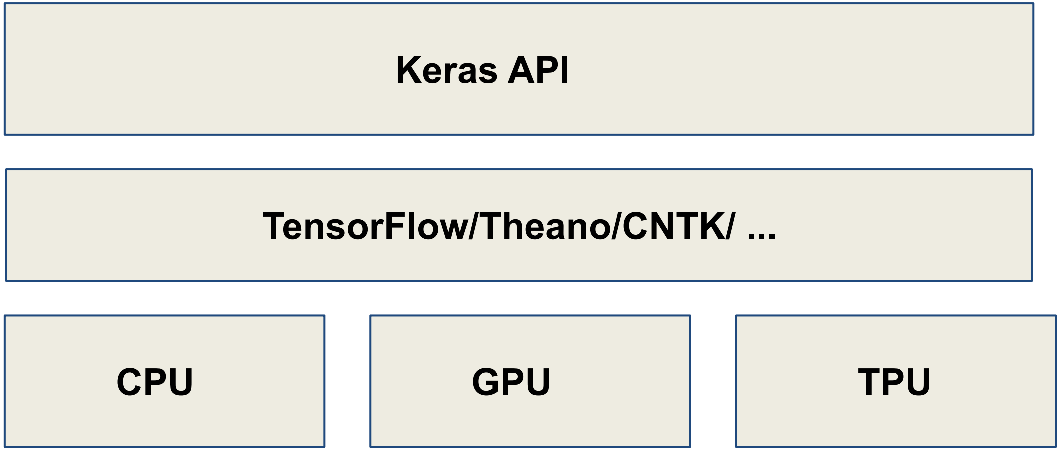
Keras is a high-level API to build and train deep learning models. It's used for fast prototyping, advanced research, and production, with four key advantages:

- *User friendly*
  Keras has a simple, consistent interface optimized for common use cases. It provides clear and actionable feedback for user errors.
- *Modular and composable*
  Keras models are made by connecting configurable building blocks together, with few restrictions.
- *Easy to extend*
  Write custom building blocks to express new ideas for research. Create new layers, loss functions, and develop state-of-the-art models.
- Distributed, multi-GPU, and TPU training

Standalone Keras

import keras

This is called Standalone Keras and it is an open source project that supports TensorFlow, Theano and CNTK backends .



Keras API

TensorFlow/Theano/CNTK/ ...

CPU

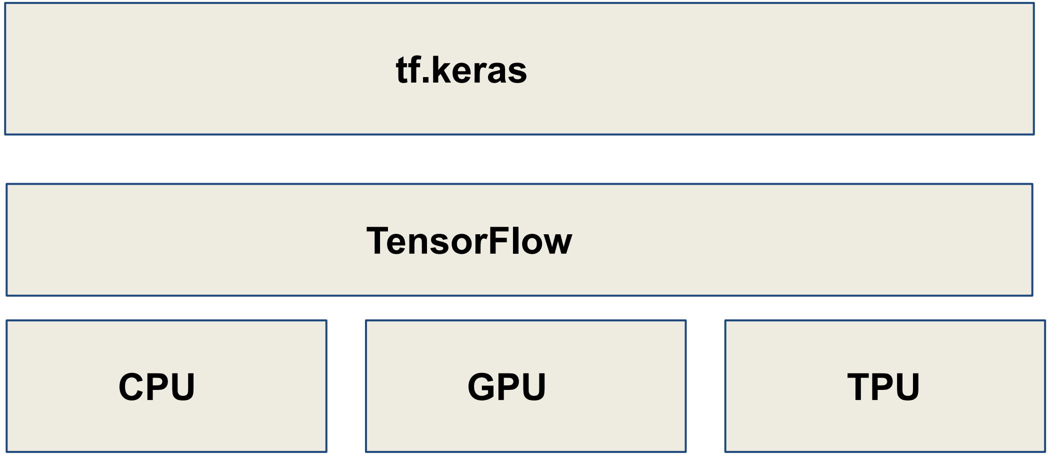
GPU

TPU

Tf.keras: official High level API of TensorFlow

tf.keras is TensorFlow’s implementation of the Keras API specification.

- Part of core TensorFlow since v1.4
- Better integration with TF-specific features like Estimator API , Eager execution etc.



API Styles

There are three styles which are used to develop the deep learning models as follows:

1) **Sequential API**

   It is used for single-input, single-output, sequential layer stacks - Good for 70+% of use cases

2) **Functional API**

   It is used for multi-input, multi-output, arbitrary static graph topologies - Good for 95% of use cases

3) **Model subclassing**

   Larger potential error surface

Note:- We will be discussing Sequential API using standalone keras in this case study

Keras important functions

First of all, Let’s import the keras standalone library using the below statement

import keras

Keras is used to create Neural network. Therefore, we need to import the layers

From keras import layers

This function is called the sequential API and is used to create the sequential model

model=keras.sequential()

This function is used to create a hidden layer and it takes many arguments (activation function, input shape, units etc) but the most important one is the number of hidden units

h1=layers.Dense(20)

This function is used to add the layer into the model

model.add(h1)

This function is used to start training the model

model.fit(X_train,labels_train)   # X and labels are the training data.

Case Study – Linear regression using TF2.0

- In this case study, we will build a very simple linear regression model from scratch and will try to predict car prices for our test set.
- We will use the functions/operators we have discussed as will see how to work with TensorFlow for model building.
- We will also discuss the Keras implementation of the same model.

Questions?