# Getting Started with TensorFlow Using Python

By Achinike Chigozie

For intermediate Python developers ready to dive into TensorFlow

**WHAT IS TENSORFLOW?**

[TensorFlow](https://www.tensorflow.org/) is an open-source end-to-end machine learning library for preprocessing data, modelling data and serving models (getting them into the hands of others).

**WHY USE TENSORFLOW?**

Rather than building machine learning and deep learning models from scratch, it's more likely you'll use a library such as TensorFlow. This is because it contains many of the most common machine learning functions you'll want to use.

## Setting Up Your Environment

We begin by importing the required libraries. TensorFlow is our main library for building models, and NumPy is used for numerical operations. We also print the TensorFlow version to make sure we're using TensorFlow 2.x, which supports eager execution by default.

# Import TensorFlow

import tensorflow as tf

print(tf.\_\_version\_\_)

## What Are Tensors?

Tensors are the core data structures in TensorFlow. They are similar to NumPy arrays but come with added features like GPU acceleration and automatic differentiation. Below are examples of how to create scalars, vectors, matrices, and higher-dimensional tensors.

**scalar = tf.constant(7)**

**vector = tf.constant([10, 10])**

**matrix = tf.constant([[7, 8], [9, 10]])**

**tensor = tf.constant([[[1, 2, 3], [4, 5, 6]],**

**[[7, 8, 9], [10, 11, 12]]])**

**print(scalar.ndim)**

**print(vector.shape)**

**print(matrix.dtype)**

## Tensor Operations

TensorFlow supports arithmetic operations such as addition, multiplication, and matrix multiplication. Here's how we can use these operations with tensors.

**a = tf.constant([[1, 2], [3, 4]])**

**b = tf.constant([[5, 6], [7, 8]])**

**tf.add(a, b)**

**tf.multiply(a, b)**

**tf.matmul(a, b)**

## Matrix Multiplication Example

Matrix multiplication is one of the most important operations in deep learning. It is used extensively in neural networks for weight calculations. Here's a practical example:

**c = tf.constant([[1, 2],**

**[3, 4]])**

**d = tf.constant([[2, 1],**

**[0, 1]])**

**print(tf.matmul(c, d))**

## Reshaping Tensors

Sometimes, we need to reshape tensors to meet the input requirements of machine learning models. TensorFlow provides reshaping functions like `reshape`, `squeeze`, and `expand\_dims` to help with this.

**tensor = tf.constant([[10, 7], [3, 4]])**

**reshaped = tf.reshape(tensor, shape=(1, 4))**

**squeezed = tf.squeeze(tf.constant([[[0], [1], [2]]]))**

**expanded = tf.expand\_dims([1, 2], axis=-1)**

## Casting and Data Types

Casting is the process of converting one data type to another. TensorFlow allows us to change the type of a tensor, which is useful when models expect a specific type (e.g., float or int).

**float\_tensor = tf.cast(tf.constant([1.7, 7.4]), dtype=tf.int32)**

**bool\_tensor = tf.cast(tf.constant([1, 0]), dtype=tf.bool)**

## Reductions and Aggregates

TensorFlow provides built-in functions to reduce tensors down to summaries. This includes computing the minimum, maximum, mean, and sum of tensor values.

**tensor = tf.constant([[10, 7], [3, 4]])**

**tf.reduce\_min(tensor)**

**tf.reduce\_max(tensor)**

**tf.reduce\_mean(tensor)**

**tf.reduce\_sum(tensor)**

## Element-wise Operations and Argmax

Element-wise operations allow us to manipulate each item in a tensor independently. `argmax` helps identify the index of the largest value, useful in classification problems.

**tensor = tf.constant([10, 7, 3])**

**tf.math.square(tensor)**

**tf.math.sqrt(tf.cast(tensor, tf.float32))**

**tf.argmax(tensor)**

## One-Hot Encoding

One-hot encoding is commonly used for categorical data. Each label is converted into a binary vector with only one bit turned on (1).

labels = [0, 1, 2, 3]

one\_hot = tf.one\_hot(labels, depth=4)

tf.one\_hot(labels, depth=4, on\_value="🔥", off\_value="❄️")

## Recap

In this foundational tutorial, you learned how to work with tensors in TensorFlow, perform basic operations, reshape data, and apply encoding techniques. These are essential skills for building machine learning models.