**Lab Exercise 1- Intro to TensorFlow & Keras Core Functions**

import tensorflow as tf

scalar = tf.constant(10)

vector = tf.constant([1, 2, 3])

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

print(scalar.shape, scalar.dtype)

print(vector.shape, vector.dtype)

print(matrix.shape, matrix.dtype)

result = tf.add(vector, [5, 5, 5])

print(result)

**Expected Output (sample):**

() <dtype: 'int32'>

(3,) <dtype: 'int32'>

(2, 2) <dtype: 'int32'>

tf.Tensor([6 7 8], shape=(3,), dtype=int32)

**Exercise 2: Using GPU**

print("GPUs Available:", len(tf.config.list\_physical\_devices('GPU')))

**Expected Output (depends on system):**

GPUs Available: 1

(or 0 if no GPU is available).

**Exercise 3: Simple Neural Network (Sequential API)**

from tensorflow import keras

from tensorflow.keras import layers

(x\_train, y\_train), (x\_test, y\_test) = keras.datasets.mnist.load\_data()

x\_train = x\_train.reshape(-1, 28\*28) / 255.0

x\_test = x\_test.reshape(-1, 28\*28) / 255.0

model = keras.Sequential([

layers.Dense(128, activation='relu', input\_shape=(784,)),

layers.Dense(64, activation='relu'),

layers.Dense(10, activation='softmax')

])

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

model.fit(x\_train, y\_train, epochs=3, batch\_size=32, validation\_split=0.1)

test\_loss, test\_acc = model.evaluate(x\_test, y\_test)

print("Test accuracy:", test\_acc)

**Expected Output (abridged):**

Epoch 1/3

1688/1688 [==============================] - 6s 3ms/step - loss: 0.3451 - accuracy: 0.9009 - val\_loss: 0.1478 - val\_accuracy: 0.9577

Epoch 2/3

1688/1688 [==============================] - 5s 3ms/step - loss: 0.1532 - accuracy: 0.9543 - val\_loss: 0.1064 - val\_accuracy: 0.9695

Epoch 3/3

1688/1688 [==============================] - 5s 3ms/step - loss: 0.1123 - accuracy: 0.9666 - val\_loss: 0.0979 - val\_accuracy: 0.9713

313/313 [==============================] - 0s 1ms/step - loss: 0.0921 - accuracy: 0.9737

Test accuracy: 0.9737

**Exercise 4: Functional API**

inputs = keras.Input(shape=(784,))

x = layers.Dense(128, activation='relu')(inputs)

x = layers.Dropout(0.2)(x)

outputs = layers.Dense(10, activation='softmax')(x)

model = keras.Model(inputs=inputs, outputs=outputs)

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

model.fit(x\_train, y\_train, epochs=3, batch\_size=32)

**Expected Output (abridged):**

Epoch 1/3

1875/1875 [==============================] - 6s 3ms/step - loss: 0.3481 - accuracy: 0.9025

Epoch 2/3

1875/1875 [==============================] - 6s 3ms/step - loss: 0.1660 - accuracy: 0.9515

Epoch 3/3

1875/1875 [==============================] - 5s 3ms/step - loss: 0.1240 - accuracy: 0.9638

**Exercise 5: Saving & Loading Models**

# Save model

model.save("my\_model.h5")

# Load model

loaded\_model = keras.models.load\_model("my\_model.h5")

print("Loaded model accuracy:", loaded\_model.evaluate(x\_test, y\_test)[1])

**Expected Output:**

313/313 [==============================] - 0s 1ms/step - loss: 0.0921 - accuracy: 0.9737

Loaded model accuracy: 0.9737