Practical – 9

Name: Shivam Tawari

Roll no: A-58

Aim: Understanding the basics of TensorFlow and Keras

Theory:

Tensorflow:

TensorFlow is an open-sourced library that's available on GitHub. It is one of the more famous libraries when it comes to dealing with Deep Neural Networks. The primary reason behind the popularity of TensorFlow is the sheer ease of building and deploying applications using TensorFlow.

TensorFlow Usage:

TensorFlow excels at numerical computing, which is critical for deep learning. It provides APIs in most major languages and environments needed for deep learning projects: Python, C, C++, Rust, Haskell, Go, Java, Android, IoS, Mac OS, Windows, Linux, and Raspberry Pi.

Moreover, TensorFlow was created keeping the processing power limitations in mind. Implying, we can run this library on all kinds of computers, irrespective of their processing powers. It can even be run on a smartphone.

Keras:

Keras is a high-level library that's built on top of Theano or TensorFlow. It provides a scikit-learn type API (written in Python) for building Neural Networks. Developers can use Keras to quickly build neural networks without worrying about the mathematical aspects of tensor algebra, numerical techniques, and optimization methods.

The key idea behind the development of Keras is to facilitate experimentations by fast prototyping. The ability to go from an idea to result with the least possible delay is key to good research.

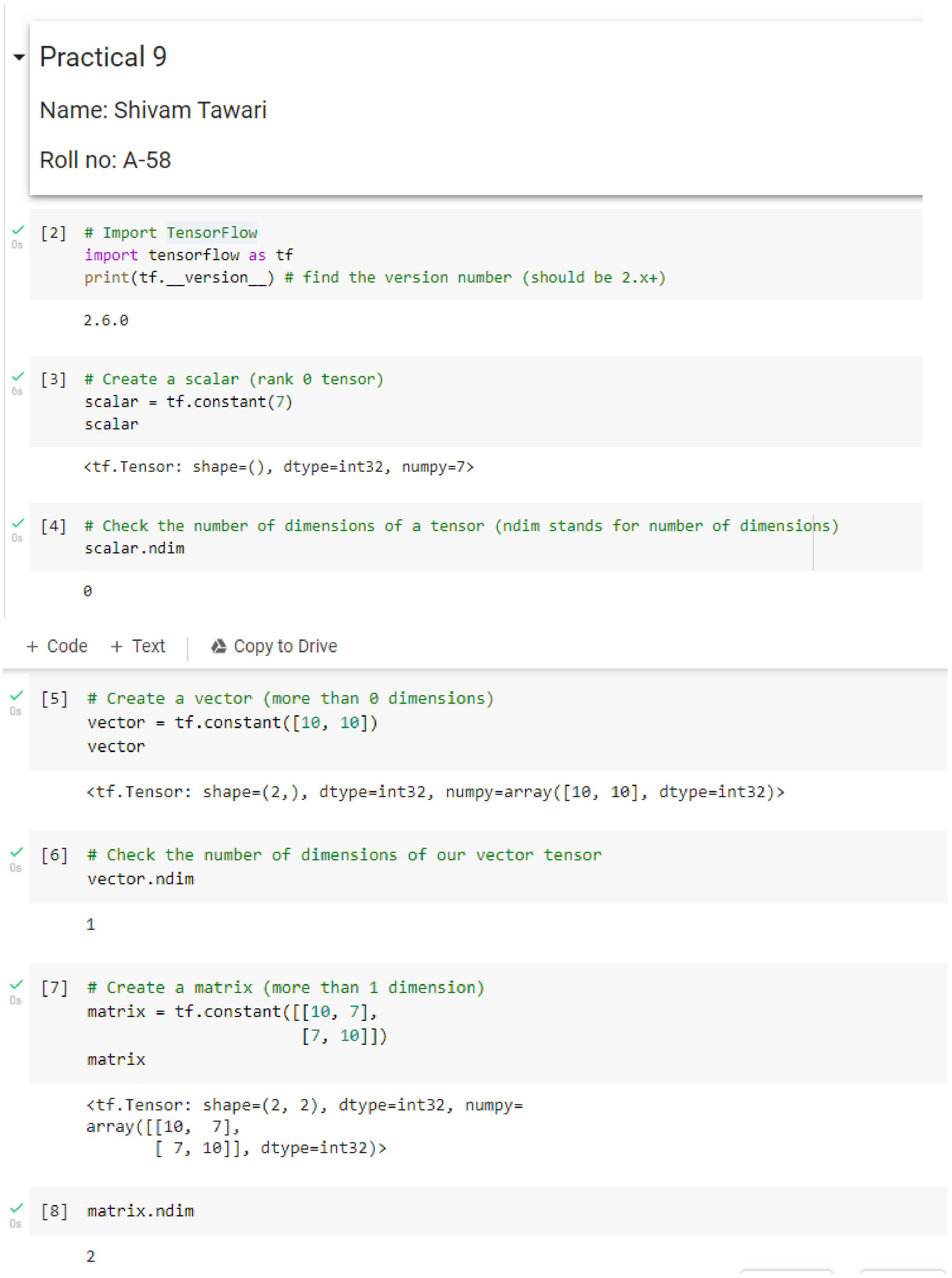
This offers a huge advantage for scientists and beginner developers alike because they can dive right into Deep Learning without getting their hands dirty with low-level computations. The rise in the demand for Deep Learning has resulted in the rise in demand for people skilled in Deep Learning.

Keras features:

- Keras is a high-level interface and uses Theano or Tensorflow for its backend.
- It runs smoothly on both CPU and GPU.
- Keras supports almost all the models of a neural network – fully connected, convolutional, pooling, recurrent, embedding, etc. Furthermore, these models can be combined to build more complex models.
- Keras, being modular in nature, is incredibly expressive, flexible, and apt for innovative research.
- Keras is a completely Python-based framework, which makes it easy to debug and explore.

Code:

TensorFlow:



Practical 9

Name: Shivam Tawari

Roll no: A-58

```
[2]  # Import TensorFlow
     import tensorflow as tf
     print(tf.__version__) # find the version number (should be 2.x+)
```

```
2.6.0
```

```
[3]  # Create a scalar (rank 0 tensor)
     scalar = tf.constant(7)
     scalar
```

```
<tf.Tensor: shape=(), dtype=int32, numpy=7>
```

```
[4]  # Check the number of dimensions of a tensor (ndim stands for number of dimensions)
     scalar.ndim
```

```
0
```

```
[5]  # Create a vector (more than 0 dimensions)
     vector = tf.constant([10, 10])
     vector
```

```
<tf.Tensor: shape=(2,), dtype=int32, numpy=array([10, 10], dtype=int32)>
```

```
[6]  # Check the number of dimensions of our vector tensor
     vector.ndim
```

```
1
```

```
[7]  # Create a matrix (more than 1 dimension)
     matrix = tf.constant([[10, 7],
                           [7, 10]])
     matrix
```

```
<tf.Tensor: shape=(2, 2), dtype=int32, numpy=
array([[10,  7],
       [ 7, 10]], dtype=int32)>
```

```
[8]  matrix.ndim
```

```
2
```

Keras:

Skills Practical - 9

Name: Shivam Tawari

Roll no: A-58

```
[3]  # first neural network with keras make predictions
     from numpy import loadtxt
     from keras.models import Sequential
     from keras.layers import Dense
```

```
[4]  # load the dataset
     dataset = loadtxt('pima-indians-diabetes.csv', delimiter=',')
     # split into input (X) and output (y) variables
     X = dataset[:,0:8]
     y = dataset[:,8]
```

```
[5]  # define the keras model
     model = Sequential()
     model.add(Dense(12, input_dim=8, activation='relu'))
     model.add(Dense(8, activation='relu'))
     model.add(Dense(1, activation='sigmoid'))
```

```
# define the keras model
model = Sequential()
model.add(Dense(12, input_dim=8, activation='relu'))
model.add(Dense(8, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

```
[6]  # compile the keras model
     model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
     # fit the keras model on the dataset
     model.fit(X, y, epochs=150, batch_size=10, verbose=0)
```

```
<keras.callbacks.History at 0x7f2ae75e7710>
```

```
# make class predictions with the model
predictions = (model.predict(X) > 0.5).astype(int)
# summarize the first 5 cases
for i in range(5):
    print('%s => %d (expected %d)' % (X[i].tolist(), predictions[i], y[i]))
```

```
[6.0, 148.0, 72.0, 35.0, 0.0, 33.6, 0.627, 50.0] => 1 (expected 1)
[1.0, 85.0, 66.0, 29.0, 0.0, 26.6, 0.351, 31.0] => 0 (expected 0)
[8.0, 183.0, 64.0, 0.0, 0.0, 23.3, 0.672, 32.0] => 1 (expected 1)
[1.0, 89.0, 66.0, 23.0, 94.0, 28.1, 0.167, 21.0] => 0 (expected 0)
[0.0, 137.0, 40.0, 35.0, 168.0, 43.1, 2.288, 33.0] => 1 (expected 1)
```

Conclusion: Hence, successfully performed Understanding the basics of TensorFlow and Keras.