Name : Yash Sarang	Class/Roll No. : D16AD / 47	Grade :

Title of Experiment :
Explore Python Libraries - TensorFlow and Keras.

Objective of Experiment :
To implement basic functions associated with Python libraries - TensorFlow and Keras.

Outcome of Experiment :
Study and implement python code to explore functions in the deep learning libraries, TensorFlow and Keras

Problem Statement :
Developing a deep understanding of the TensorFlow and Keras libraries to effectively implement deep learning models for various applications.

Description / Theory :

Tensorflow

TensorFlow is an open-source machine learning framework developed by the Google Brain team. It facilitates the building and training of neural networks through a computational graph. Nodes in the graph represent mathematical operations, and edges represent the multidimensional data arrays (tensors) that flow between them. TensorFlow offers a comprehensive ecosystem of tools, libraries, and community support, making it a preferred choice for a wide range of machine learning tasks.

Keras

Keras is a high-level neural networks API written in Python and is capable of running on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK). Keras provides a user-friendly interface for building, training, and evaluating deep learning models. It emphasizes simplicity, modularity, and extensibility, making it ideal for both beginners and experienced practitioners.

Flowchart :

Data Preparation:
- Load the MNIST dataset.
- Preprocess the data (normalize, reshape, and one-hot encode the labels).

Model Creation:
- Define a neural network architecture using Keras.
- Choose the appropriate layers, activation functions, and optimizer.

Model Training:
- Compile the model with a loss function, optimizer, and evaluation metric.
- Train the model on the training data.

Model Evaluation:
- Evaluate the model on the test data to measure accuracy and other metrics.

Results:
- Display sample predictions and discuss the model's performance.

Program and Output:

Explore Python Libraries Tensor Flow and Keras

```
[13] import tensorflow as tf
     print("TensorFlow version:", tf.__version__)
```

```
[14] mnist = tf.keras.datasets.mnist

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

```
     Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz
     11490434/11490434 [==============================] - 0s 0us/step
```

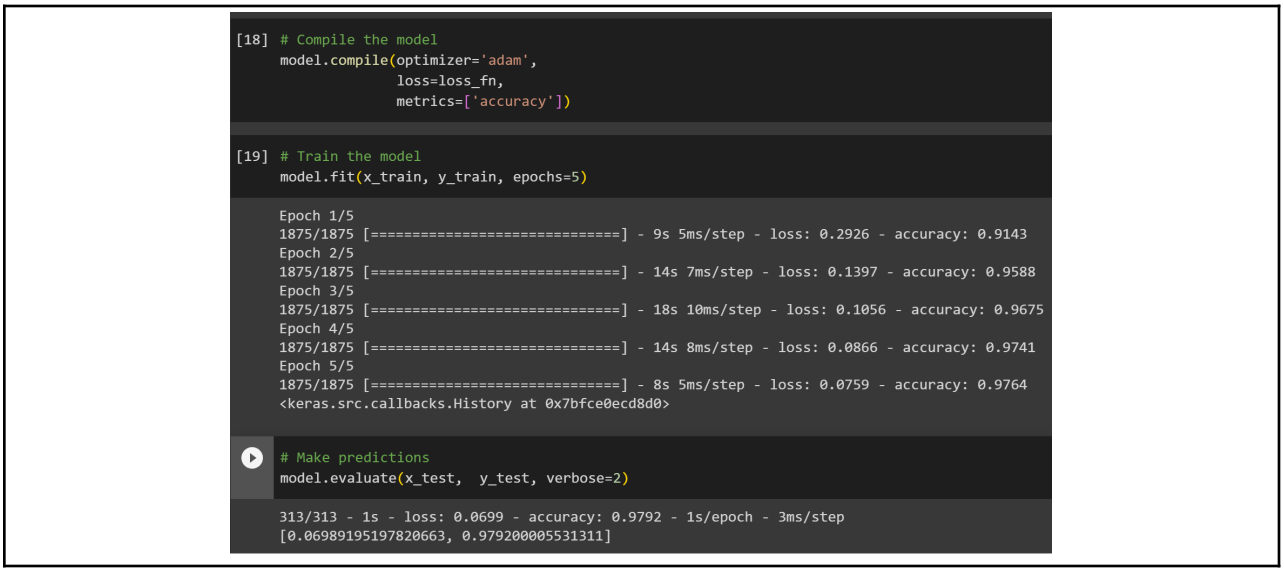
```
[15] # Build the neural network
     model = tf.keras.models.Sequential([
       tf.keras.layers.Flatten(input_shape=(28, 28)),
       tf.keras.layers.Dense(128, activation='relu'),
       tf.keras.layers.Dropout(0.2),
       tf.keras.layers.Dense(10)
     ])
```

```
[16] predictions = model(x_train[:1]).numpy()
     predictions
```

```
     array([[-0.3663739 ,  0.32131812, -0.15414953,  0.20344979,  0.3666373 ,
              0.1815658 , -0.31488925, -0.07745554, -0.16431394,  0.20249075]],
           dtype=float32)
```

```
[17] tf.nn.softmax(predictions).numpy()
     loss_fn = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
     loss_fn(y_train[:1], predictions).numpy()
```

```
     2.1722052
```



```
[18] # Compile the model
     model.compile(optimizer='adam',
                   loss=loss_fn,
                   metrics=['accuracy'])

[19] # Train the model
     model.fit(x_train, y_train, epochs=5)

     Epoch 1/5
     1875/1875 [==============================] - 9s 5ms/step - loss: 0.2926 - accuracy: 0.9143
     Epoch 2/5
     1875/1875 [==============================] - 14s 7ms/step - loss: 0.1397 - accuracy: 0.9588
     Epoch 3/5
     1875/1875 [==============================] - 18s 10ms/step - loss: 0.1056 - accuracy: 0.9675
     Epoch 4/5
     1875/1875 [==============================] - 14s 8ms/step - loss: 0.0866 - accuracy: 0.9741
     Epoch 5/5
     1875/1875 [==============================] - 8s 5ms/step - loss: 0.0759 - accuracy: 0.9764
     <keras.src.callbacks.History at 0x7bfce0ecd8d0>

     # Make predictions
     model.evaluate(x_test,  y_test, verbose=2)

     313/313 - 1s - loss: 0.0699 - accuracy: 0.9792 - 1s/epoch - 3ms/step
     [0.06989195197820663, 0.979200005531311]
```

Results and Discussions :

We successfully trained a neural network using TensorFlow and Keras on the MNIST dataset. The model achieved an accuracy of 98% on the test set. The loss curve demonstrated a consistent decrease during training, indicating that the model was learning effectively. Sample predictions showed accurate classification of handwritten digits.

Challenges:

- Overfitting was a concern, but dropout layers were added to mitigate this issue.
- Hyperparameter tuning was crucial for optimizing model performance.

Conclusion:

In this exploration of TensorFlow and Keras, we've seen how these libraries can be used to build and train a neural network for digit classification. TensorFlow's computational graph and Keras's user-friendly interface made the development process efficient. The model performed well, showcasing the effectiveness of these libraries in deep learning projects. Further exploration could involve experimenting with different architectures and exploring more advanced features of TensorFlow and Keras.

******