DL_Lab_Experiment2_TensorFlow_AparnaIyer

October 23, 2024

Name: Aparna Iyer

PRN: 22070126017

Batch: 2022-2026

Branch: AI-ML A1

1 Experiment 2

1.1 1. Title: Fundamentals of TensorFlow

1.2 2. Objectives:

a. Study the fundamentals of TensorFlow.

b. Understand TensorFlow's core components, such as tensors, computational graphs, and sessions.

c. Learn how to implement a simple neural network using TensorFlow.

d. Develop a simple Python program to train a neural network for a basic classification task (e.g., MNIST dataset).

1.3 3. Theory:

TensorFlow Overview: TensorFlow is an open-source deep learning framework developed by Google. It facilitates the design, building, and training of deep learning models through a comprehensive ecosystem. TensorFlow primarily uses data flow graphs to represent computations. The nodes in these graphs represent operations, while the edges represent the data (tensors) flowing between them.

Core Concepts: 1. Tensors: The primary data structure in TensorFlow, similar to arrays or matrices in other libraries. 2. Graph: TensorFlow uses computational graphs where nodes represent operations, and edges represent the tensors passed between operations. 3. Session: The execution environment where the operations in the graph are run.

Advantages of TensorFlow:

- Scalable and efficient.
- Has support for GPUs and TPUs.
- Extensive library for building neural networks and deploying machine learning models.

Basic Neural Network Concepts:

A neural network consists of layers of interconnected neurons. Each layer applies transformations to the input data and passes it to the next layer. The learning process involves adjusting the weights and biases of these neurons to minimize a loss function. For this experiment, we'll build a simple neural network using TensorFlow to classify the digits from the MNIST dataset.

[2]:
```python
#Import the TensorFlow Library
import tensorflow as tf
from tensorflow.keras import datasets, layers, models
import matplotlib.pyplot as plt

# Load and preprocess the MNIST dataset
(x_train, y_train), (x_test, y_test) = datasets.mnist.load_data()

# Normalize the pixel values to the range 0 to 1
x_train, x_test = x_train / 255.0, x_test / 255.0

# Build a simple neural network model using TensorFlow
model = models.Sequential([
    layers.Flatten(input_shape=(28, 28)),  # Flatten the 28x28 image to a 1D␣
↪vector
    layers.Dense(128, activation='relu'),  # Fully connected layer with 128␣
↪units
    layers.Dense(10, activation='softmax')  # Output layer for 10 classes␣
↪(digits 0-9)
])

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

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

# Evaluate the model
test_loss, test_acc = model.evaluate(x_test, y_test)
print(f'\nTest accuracy: {test_acc}')
```

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

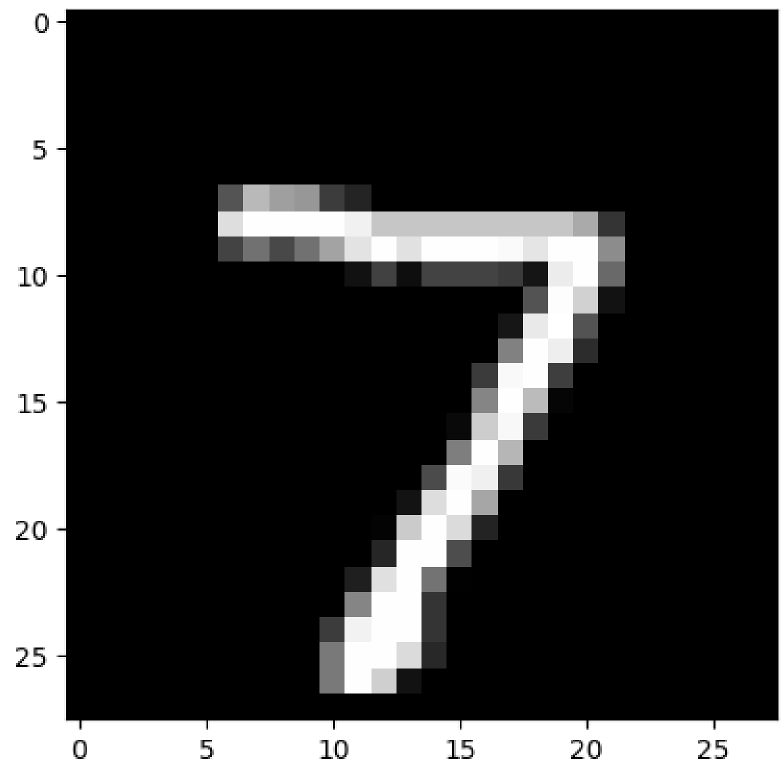
```
/usr/local/lib/python3.10/dist-
packages/keras/src/layers/reshaping/flatten.py:37: UserWarning: Do not pass an
`input_shape`/`input_dim` argument to a layer. When using Sequential models,
prefer using an `Input(shape)` object as the first layer in the model instead.
```

```
  super().__init__(**kwargs)

Epoch 1/5
1875/1875              7s 3ms/step -
accuracy: 0.8796 - loss: 0.4252
Epoch 2/5
1875/1875              14s 5ms/step -
accuracy: 0.9673 - loss: 0.1158
Epoch 3/5
1875/1875              9s 5ms/step -
accuracy: 0.9768 - loss: 0.0767
Epoch 4/5
1875/1875              8s 4ms/step -
accuracy: 0.9837 - loss: 0.0551
Epoch 5/5
1875/1875              10s 4ms/step -
accuracy: 0.9875 - loss: 0.0425
313/313            1s 1ms/step -
accuracy: 0.9735 - loss: 0.0826

Test accuracy: 0.9771000146865845
```

[5]:

```python
# Visualize the first test image and prediction
import numpy as np #Importing the NumPy Library

plt.imshow(x_test[0], cmap='gray') #Displaying the Input in Grayscale
predicted_class = np.argmax(model.predict(x_test[:1]), axis=1)[0]
print(f"Predicted: {predicted_class}")
plt.show()
```

```
1/1            0s 18ms/step
Predicted: 7
```



1.4 4. Conclusion:

In this experiment, we explored the basics of TensorFlow and its core components. We successfully implemented a neural network model to classify the MNIST dataset.

The model achieved good accuracy, demonstrating how TensorFlow can be used to build and train deep learning models efficiently.

Through this process, we gained insights into TensorFlow's functionality, including model building, compilation, and evaluation.

[ ]:
```
!apt-get install texlive texlive-xetex texlive-latex-extra pandoc
!pip install pypandoc
```

[ ]:
```
from google.colab import drive
drive.mount('/content/drive')
```

[18]:
```
!jupyter nbconvert --to PDF "/content/drive/MyDrive/Colab Notebooks/
↪DL_Lab_Experiment2_TensorFlow_AparnaIyer.ipynb"
```

```
[NbConvertApp] Converting notebook /content/drive/MyDrive/Colab
Notebooks/DL_Lab_Experiment2_TensorFlow_AparnaIyer.ipynb to PDF
[NbConvertApp] Support files will be in
DL_Lab_Experiment2_TensorFlow_AparnaIyer_files/
```

```
[NbConvertApp] Making directory ./DL_Lab_Experiment2_TensorFlow_AparnaIyer_files
[NbConvertApp] Writing 57051 bytes to notebook.tex
[NbConvertApp] Building PDF
[NbConvertApp] Running xelatex 3 times: ['xelatex', 'notebook.tex', '-quiet']
[NbConvertApp] Running bibtex 1 time: ['bibtex', 'notebook']
[NbConvertApp] WARNING | bibtex had problems, most likely because there were no
citations
[NbConvertApp] PDF successfully created
[NbConvertApp] Writing 73089 bytes to /content/drive/MyDrive/Colab
Notebooks/DL_Lab_Experiment2_TensorFlow_AparnaIyer.pdf
```