4.10. Experiment No. C-2

Aim: Implementation of MNIST Handwritten Character Detection using PyTorch, Keras and Tensorflow

Theory:

The MNIST handwritten digit classification problem is a standard dataset used in computer vision and deep learning. Although the dataset is effectively solved, it can be used as the basis for learning and practicing how to develop, evaluate, and use convolutional deep learning neural networks for image classification from scratch. This includes how to develop a robust test harness for estimating the performance of the model, how to explore improvements to the model, and how to save the model and later load it to make predictions on new data. MNIST is a widely used dataset of handwritten digits that contains 60,000 handwritten digits for training a machine learning model and 10,000 handwritten digits for testing the model. It was introduced in 1998 and has become a standard benchmark for classification tasks. It is also called the “Hello, World” dataset as it’s very easy to use. MNIST was derived from an even larger dataset, the NIST Special Database 19 which not only contains digits but also uppercase and lowercase handwritten letters. In the MNIST dataset each digit is stored in a grayscale image with a size of 28x28 pixels.

Tensorflow: Tensorflow is an open-source library, and we use TensorFlow to train and develop machine learning models.

Keras: It is also an open-source software library and a high-level TensorFlow API. It also provides a python interface for Artificial Neural Networks.

Pytorch: An open-source ML framework based on Python Programming Language and torch Library.

Implementation:

Dataset: To build this application, we use the MNIST dataset. This dataset contains images of digits from 0 to 9. All these images are in greyscale. There are both training images and testing images. This dataset contains about 60000 training images which are very large and about 10000 testing images. All these images are like small squares of 28 x 28 pixels in size. These are handwritten images of individual digits. Importing Libraries: Import all the required libraries before writing any code. In the beginning, | had already mentioned all the requirements for building the application. So import those libraries. From PIL library import ImageGrab and Image.

Building Model using Tensorflow: To build the model first we need to import some libraries from TensorFlow Keras. We have to import keras from TensorFlow and then import the dataset that we are going to use to build our application now. That is the MNIST dataset. Then import the sequential model, and some layers like Dense, Dropout, Flatten Conv2D, MaxPooling2D, and finally import the backend. After importing all the required libraries, split the dataset into train and test datasets. Reshape training set of x and testing set of x. The next step is to convert class vectors to binary class matrices.

Training the Model on Tensorflow: Our next step is to train the model. For that define batch size, the number of classes, and the number of epochs that you want to train your model. next add some layers to the sequential model which we imported before. Then compile the model using categorical cross-entropy loss function, Adadelta optimizer, and accuracy metrics. Finally using x_train,y_train, batch size, epochs, and all train the model. Then save it for later.

Predicting Digit: Now we have to write some code for predicting the digit that we have written. For that define a function predict_class where you provide an image as an argument. First, resize it into the required pixels. Convert the image into grayscale which was previously in RGB. Then reshape and normalize it.

Finally, predict the image using predict method. Building Application: Let us see how to build a user-friendly GUI application for it. We use Tkinter for it. Here we create some space for the user to actually draw the digit and then provide two buttons Recognize and clear. Recognize button is to recognize the digit that is written on the given space and the clear button is to clear the writings on it. Finally, run the main loop to run the application. Observation: When you run the application a window will pop up where you can write the digit. And next, when you click on recognize button, it will recognize the digit you have written with the probability percentage showing how exactly the digit matches with the original one. Here | have written digit 1 and it recognized it as 1 with 17% accuracy.

Input:

Output: [5]

Conclusion: We have successfully implemented MNIST Handwritten Character Detection using PyTorch, Keras and Tensorflow.

Experiment Level Outcome (ELO 1): Student will be able to understand and implement MNIST Handwritten Character Detection using PyTorch, Keras and Tensorflow

Questions:

1. What is the purpose of handwritten digit recognition?

2. What is the problem with handwritten recognition?

3. What neural network is used for handwritten digit recognition?

4. Which dataset is used for handwritten digit recognition?

5. What is the full form of MNIST?