Name: Aziz Sayyad
Roll No: 381069
Batch: A3

Assignment No: - 1

Problem Statement:
Implementing a Feedforward Neural Network in Python using Keras and TensorFlow for handwritten digit classification on the Arabic Handwritten Digits Dataset.

Objective:

- To understand the structure of feedforward neural networks.
- To preprocess handwritten digit data for model training.
- To implement a feedforward neural network using Keras and TensorFlow.
- To evaluate the model's classification performance using validation data.
- To visualize training and validation loss across epochs.

S/W Packages and H/W apparatus used:

- **Operating System:** Windows/Linux/MacOS
- **Kernel:** Python 3.x
- **Tools:** Jupyter Notebook, Anaconda, or Google Colab
- **Hardware:** CPU with minimum 4GB RAM; GPU optional for faster execution
- **Libraries and packages used:** TensorFlow, Keras, NumPy, Pandas, Matplotlib, Scikit-Learn

Theory:
A Feedforward Neural Network (FNN) is a type of artificial neural network where data flows only in one direction — from the input layer, through hidden layers, and finally to the output layer. Unlike recurrent neural networks, it has no feedback connections or cycles.

Structure:

- **Input Layer:** Takes digit image features (28×28 pixels).
- **Hidden Layers:** Perform computations with dense connections and non-linear activations.
- **Output Layer:** Produces probabilities for each digit (0–9).
- **Activation Functions:** ReLU for hidden layers, Softmax for output layer.
- **Backpropagation:** Used to train the model by minimizing loss through weight updates

Methodology:

1. **Data Acquisition:** Load the Arabic Handwritten Digits dataset from CSV files for images and labels.
2. **Data Preparation:** Normalize pixel values between 0 and 1; apply one-hot encoding to labels.
3. **Model Architecture:** Create a sequential model with layers:
   - Reshape input (28×28) into a 1D vector.
   - Dense(256, ReLU).
   - Dense(192, ReLU).
   - Dense(128, ReLU).
   - Dense(10, Softmax) for classification.
4. **Model Compilation:** Use Adam optimizer and categorical crossentropy loss.
5. **Model Training:** Train the model on the training data, validate on the test set, and track performance metrics.
6. **Model Evaluation:** Predict on the test set, compare predictions with true labels using accuracy and confusion matrix.
7. **Loss Visualization:** Plot training and validation loss to assess learning behavior.

Advantages:

- Handles non-linear data patterns through activation functions.
- Easily adaptable to classification tasks with multiple classes.
- Scalable with deeper layers and more neurons for higher accuracy.
- Capable of generalizing well when trained with sufficient data.
- Supports GPU-based parallel processing for faster computation.

Limitations:

- Requires a large amount of labeled image data for effective training.
- High computational cost for training deep architectures.
- Functions as a black-box model with limited interpretability.
- Prone to overfitting if not regularized properly.
- Performance is highly dependent on hyperparameter tuning.

Applications:

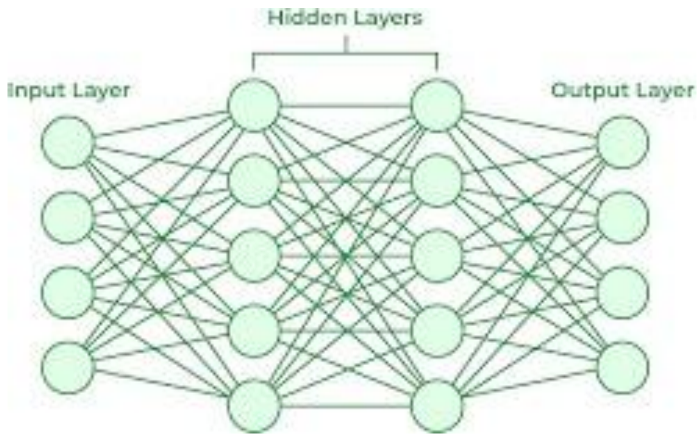
- Digit recognition systems (e.g., postal code detection, bank cheque processing).
- Optical Character Recognition (OCR).
- Handwriting-based authentication systems.
- Automated data entry and form processing.
- Educational tools for digit recognition practice.

Working / Algorithm:

1. Import required libraries (NumPy, Pandas, TensorFlow, etc.).
2. Load dataset (training and testing images/labels).
3. Visualize sample digits from dataset.
4. Normalize pixel values to range (0–1).
5. Apply one-hot encoding to labels.
6. Reshape input images for compatibility with Dense layers.
7. Build the model with dense layers and softmax output.

8. Compile the model using Adam optimizer and categorical crossentropy loss.
9. Train the model while validating with test data.
10. Evaluate using accuracy, confusion matrix, and classification report.
11. Plot training and validation loss to visualize learning progress.
12. Deploy the model for handwritten digit predictions.

Diagram:



Conclusion:
The Feedforward Neural Network (FNN) successfully classified Arabic handwritten digits by learning from pixel data. The use of ReLU activations and Softmax output enabled the model to handle non-linearity and multi-class classification effectively. Although computationally intensive and prone to overfitting, with proper preprocessing and optimization, FNNs provide a robust and scalable solution for digit recognition tasks.