Image Classification with FCNN: Task 1

Group 4:
Hardik Singh (2101MC19)
Mohd Darish Khan (2101MC29)
Suryansh Jaiswal (2101MC41)

September 17, 2024

1 Introduction

This report details the processes involved in image classification using Fully Connected Neural Networks (FCNN). The dataset consists of images categorized into five classes: butterfly, menorah, scorpion, starfish, and others. The task involves data preprocessing, building different models with regularization techniques, and evaluating the model's performance.

2 Importing Libraries

The first step involves importing the necessary libraries such as `numpy`, `pandas`, `tensorflow`, and `matplotlib` for data manipulation, model building, and visualization. These libraries are critical for handling image data and performing the classification task.

3 Extracting and Preprocessing Data

3.1 Data Extraction

The image dataset is extracted from a zip file. Each image is resized to a uniform shape of 224x224 pixels, and its corresponding label is assigned. The dataset is divided into three parts: training, validation, and testing.

- Training set: 250 images
- Validation set: 50 images
- Test set: 100 images

3.2 Data Preprocessing

The images are converted into NumPy arrays and normalized by scaling the pixel values to a range of 0 to 1. Grayscale images are converted to RGB by repeating the grayscale channel three times. Labels are also converted into numerical format for model training.

4 Model Architecture

4.1 FCNN without Regularization

The first model is a simple Fully Connected Neural Network without any regularization or dropout layers. It consists of the following layers:

- Input Layer: 224x224x3 (for RGB images)
- Flatten Layer: Converts the image into a 1D array.
- Dense Layers: Two dense layers with 128 and 64 neurons, both with ReLU activation.

- Output Layer: A softmax layer with 5 outputs corresponding to the five classes.

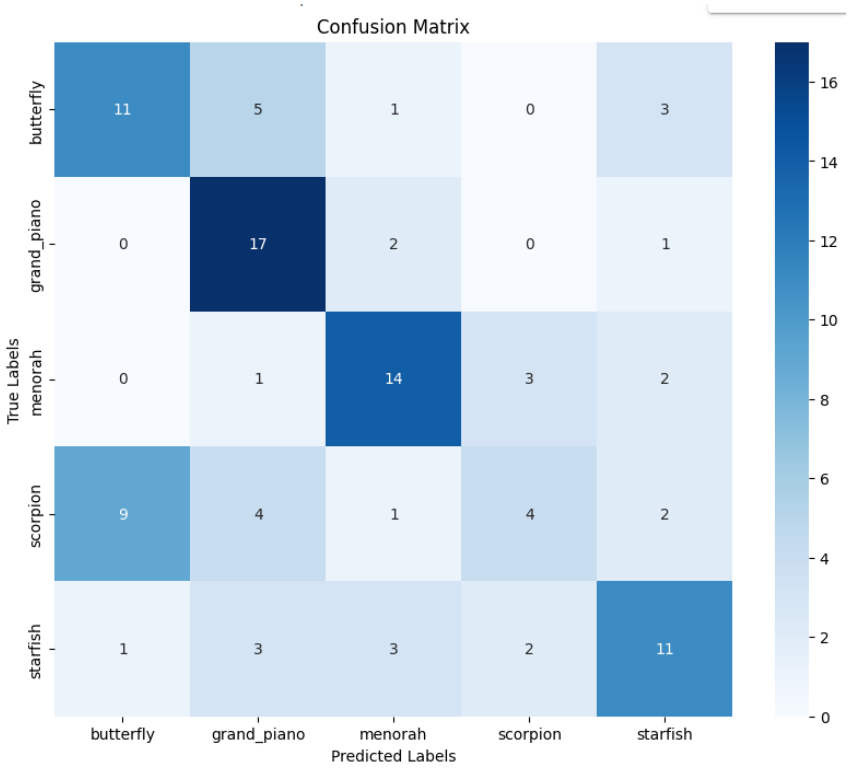

Figure 1: Confusion Matrix of FCNN without regularisation

4.2 FCNN with L2 Regularization

A second model was built incorporating L2 regularization to prevent overfitting. L2 regularization was applied to the Dense layers to penalize large weights.

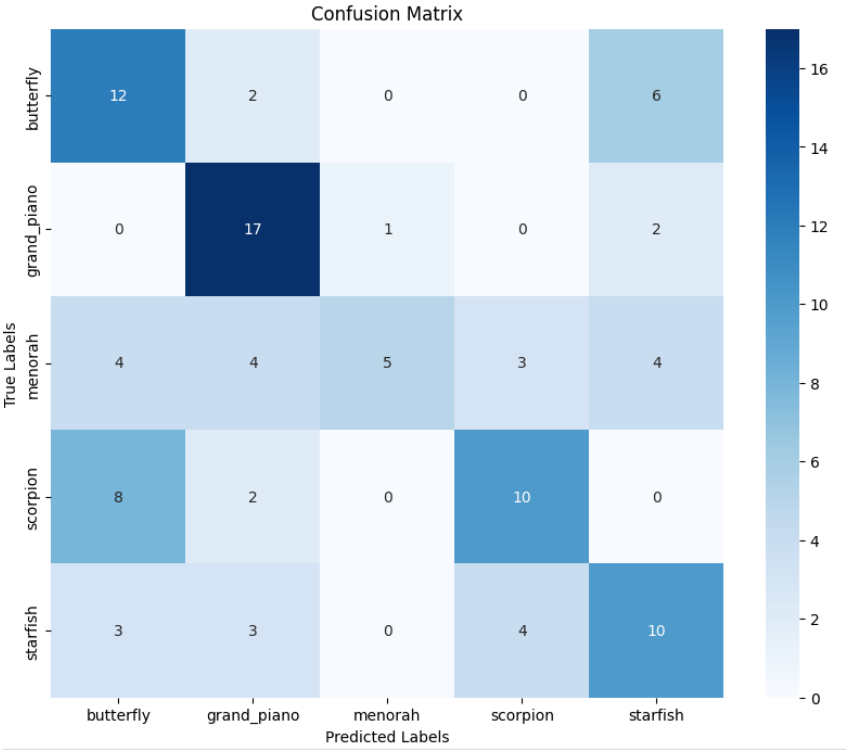

Figure 2: Confusion Matrix of FCNN with L2 regularisation

4.3 FCNN with Batch Normalization and Dropout

In the third model, both batch normalization and dropout layers were added to further improve generalization. Dropout layers help prevent overfitting by randomly setting a fraction of the input units to zero during training.

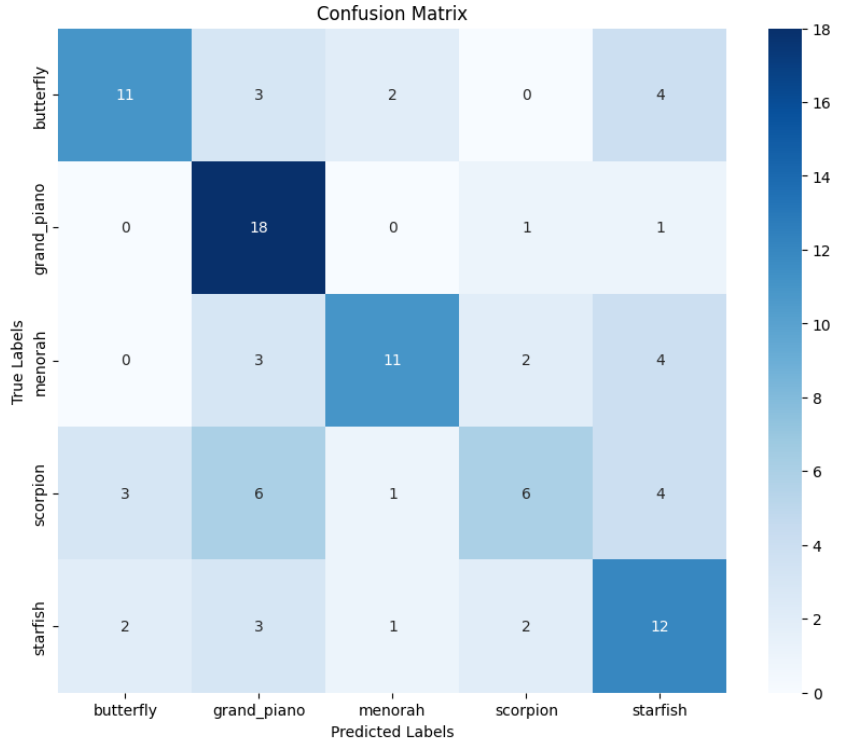

Figure 3: Confusion Matrix of FCNN without regularisation

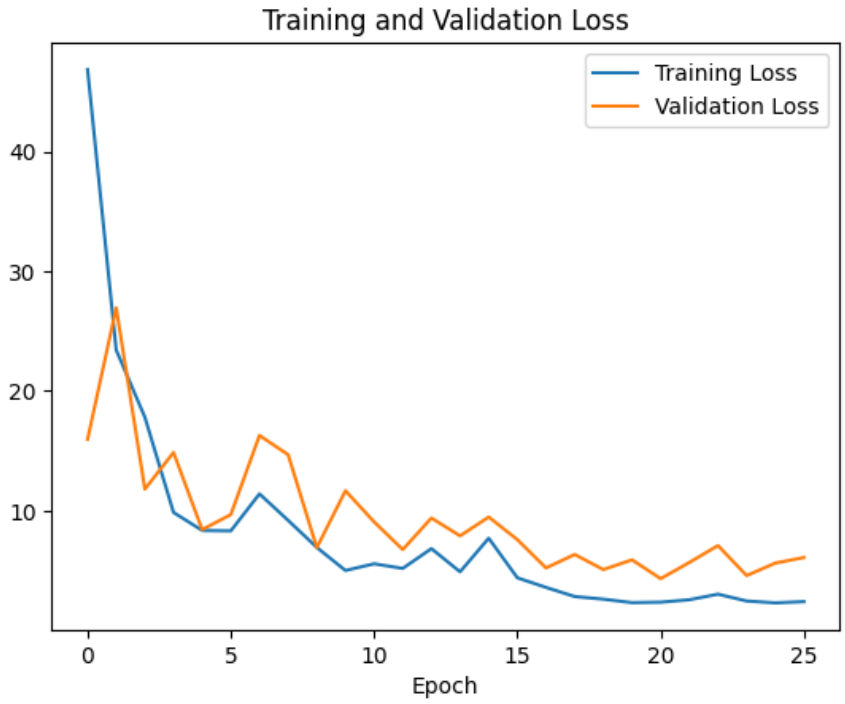

Figure 4: Training vs Validation Loss

5 Training and Evaluation

The models were trained using the `Adam` optimizer and sparse categorical cross-entropy as the loss function. Early stopping was used to prevent overfitting by monitoring validation loss and stopping the training once the loss stopped improving.

5.1 Training Accuracy

The models achieved varying levels of accuracy during training. The best performance was achieved by the model with L2 regularization, dropout, and batch normalization.

5.2 Test Accuracy and Confusion Matrix

After training, the models were evaluated on the test set, and the test accuracy was reported. The confusion matrix for the best model is presented below, showing the performance in each class.

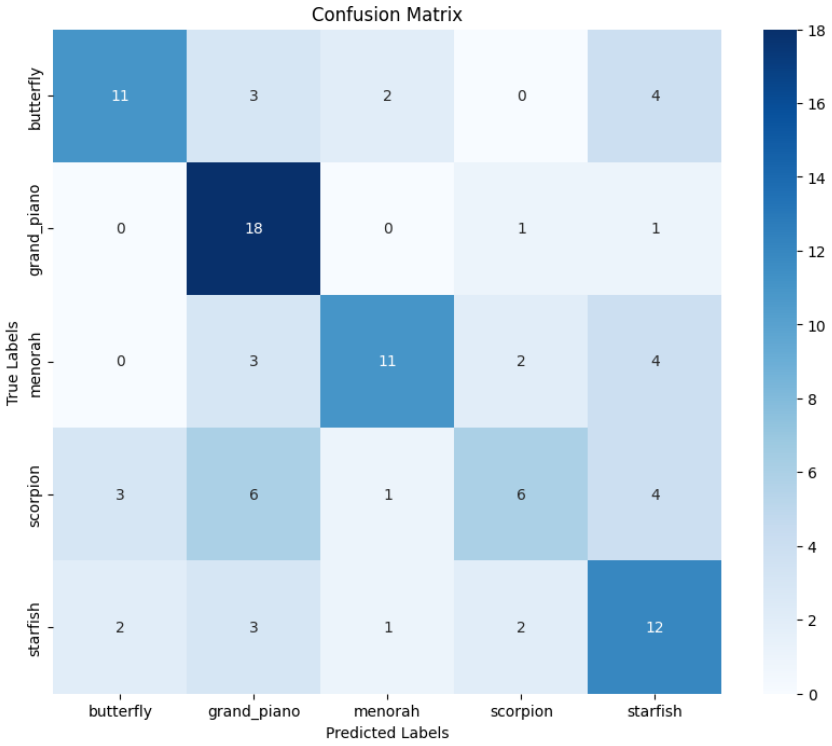

Figure 5: Confusion Matrix for the Best Model

6 Conclusion

In this assignment, we built and evaluated multiple FCNN models for image classification. Incorporating regularization techniques such as L2 regularization, batch normalization, and dropout significantly improved the model's generalization performance. Future work could involve using more advanced architectures such as Convolutional Neural Networks (CNNs) for better accuracy in image classification tasks.