Department of Electrical and Electronics Engineering

EEE 443: Neural Networks

Class Project VI Report

Name and Surname: Ali Aral Takak

Student ID: 22001758

Department: EEE

Instructor: Erdem Koyuncu

Introduction

In the sixth project of the course, a neural network was developed to classify colored images of geometric shapes into one of nine predefined categories: **Circle, Square, Octagon, Heptagon, Nonagon, Star, Hexagon, Pentagon, and Triangle.** The dataset consisted of $\mathbf{90.000}$ images, each sized $\mathbf{200 \times 200}$ pixels, and was divided into a training set with $\mathbf{8.000}$ images per class and a testing set with $\mathbf{2.000}$ images per class. The project includes preparing the data, designing a neural network architecture, and training the model to recognize shape features correctly.

Architecture Breakdown

The neural network that we have implemented for this project is a convolutional neural network designed to classify images of geometric shape into one of the nine classes. The network is composed to multiple convolutional layers, pooling layers, fully connected layers, and dropout layers. The details of the network can be stated as follows:

Input Layer: The input of the network is a single-channel image of size 200×200 pixels. The images are transformed using resizing, flipping, rotation, and brightness/contrast adjustment in order to improve the generalization property of the network.

Feature Extraction Layers: The feature extraction part of the network consists of two convolutional blocks, each followed by a Rectified Linear Unit (ReLU) activation function and batch normalization. The first convolutional block applies 32 filters of size 3×3 to the input image. The maxpooling in this layer reduces the spatial size by half, changing the resolution from 200×200 to 99×99 after padding and pooling. The second convolutional block uses 64 filters of size 3×3 to extract the more complex features from images. This block also includes batch normalization, maxpooling, and then is followed by a ReLU activation function. The output spatial size becomes 48×48 after this layer. After these two convolution blocks, three additional maxpooling layers reduce the spatial resolution by 2 each, finally reducing the resolution to 6×6.

Fully Connected Layers: The 3-dimensional output tensor is flattened into a 1-dimensional vector of size $64 \times 6 \times 6 = 2304$ per sample and passed through fully connected layers. The first fully connected layer reduces the feature vector from 2304 to 512 dimensions. This first fully connected layer is followed by a ReLU activation and a dropout regularization with a probability of 0.5. The

second connected layer reduces the representation from 512 dimensions to 128 dimensions and is followed by a ReLU activation. The output layer maps the 128-dimensional feature vector to the 9 output classes. The final classification is obtained using cross-entropy loss, which applies the softmax function internally. We also use Adam optimizer with learning rate scheduling during training for better convergence and results.

Results

With the network architecture explained above, we have executed several runs on the dataset, in which we have experimented with various values of epochs, learning rates, and batch sizes. The figures given below displays the accuracy and loss results for 25 epochs, with a learning rate of 0.001 and 64 batch size:

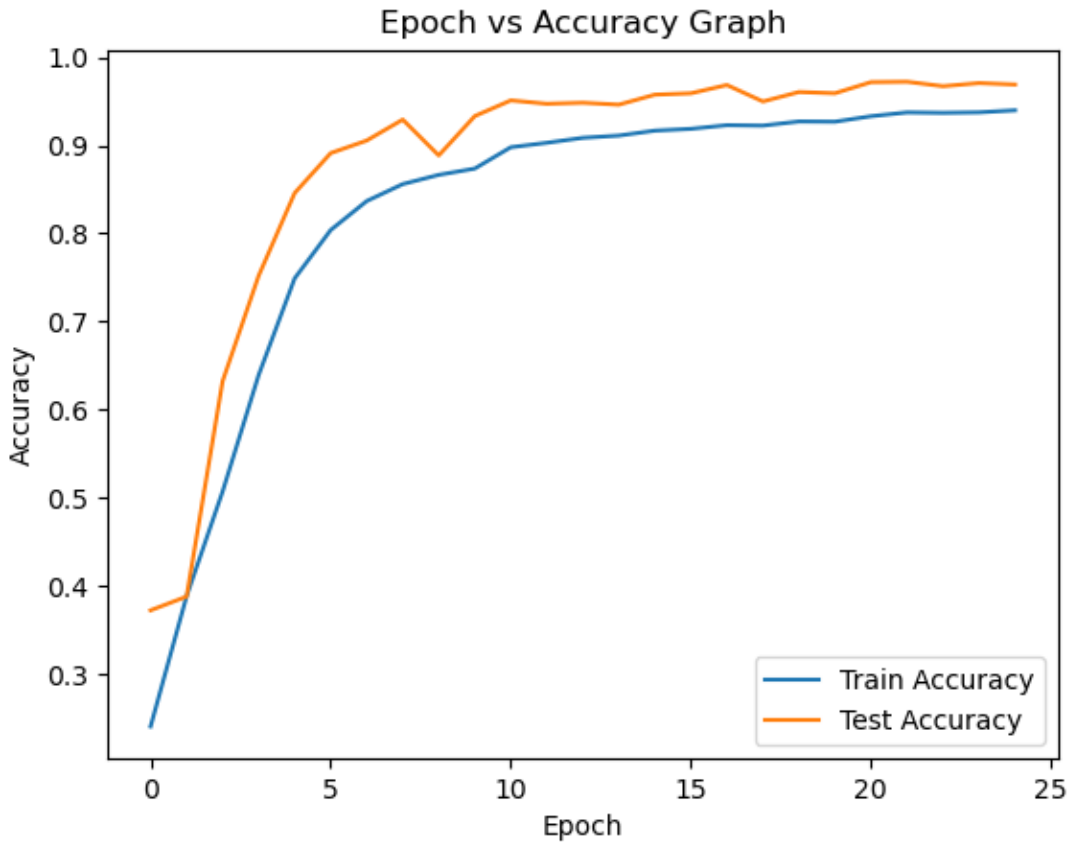

Figure 1: Epoch versus accuracy graph for epoch = 25, learning rate = 0.001, batch size = 64.

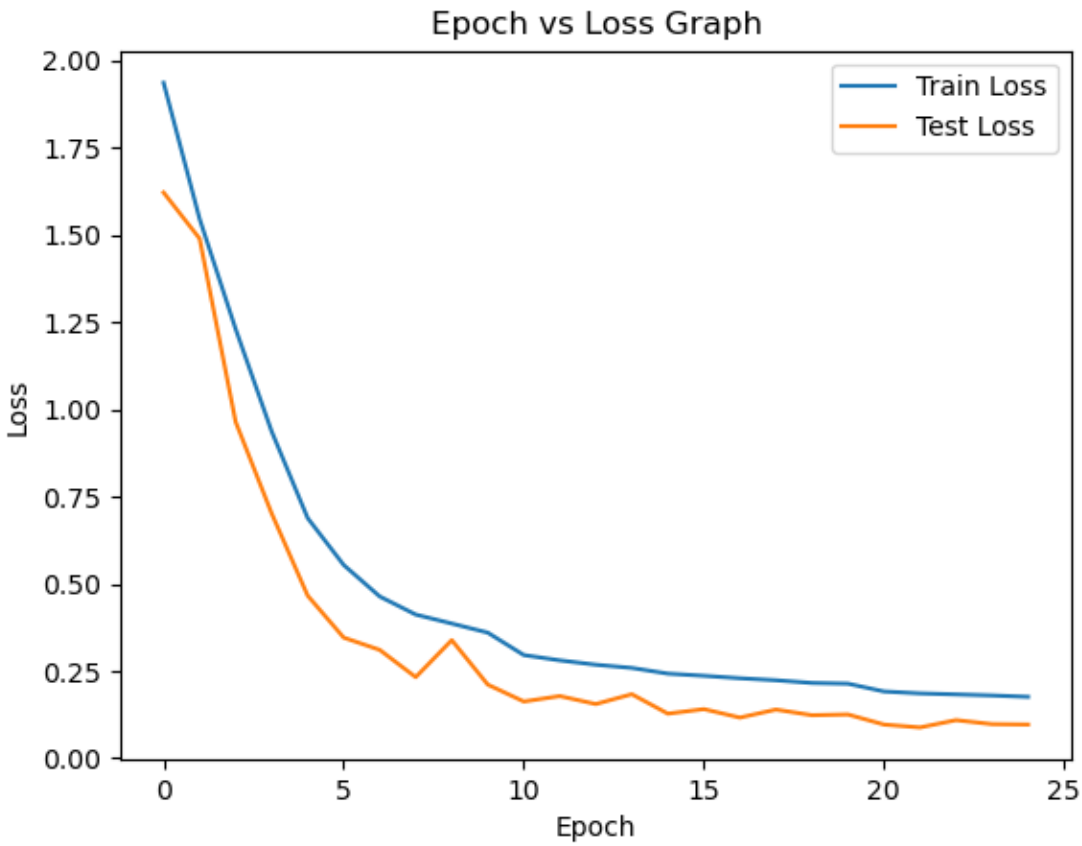

Figure 2: Epoch versus loss graph for epoch = 25, learning rate = 0.001, batch size = 64.

The figures given below displays the accuracy and loss results for 15 epochs, with a learning rate of 0.001 and 64 batch size:

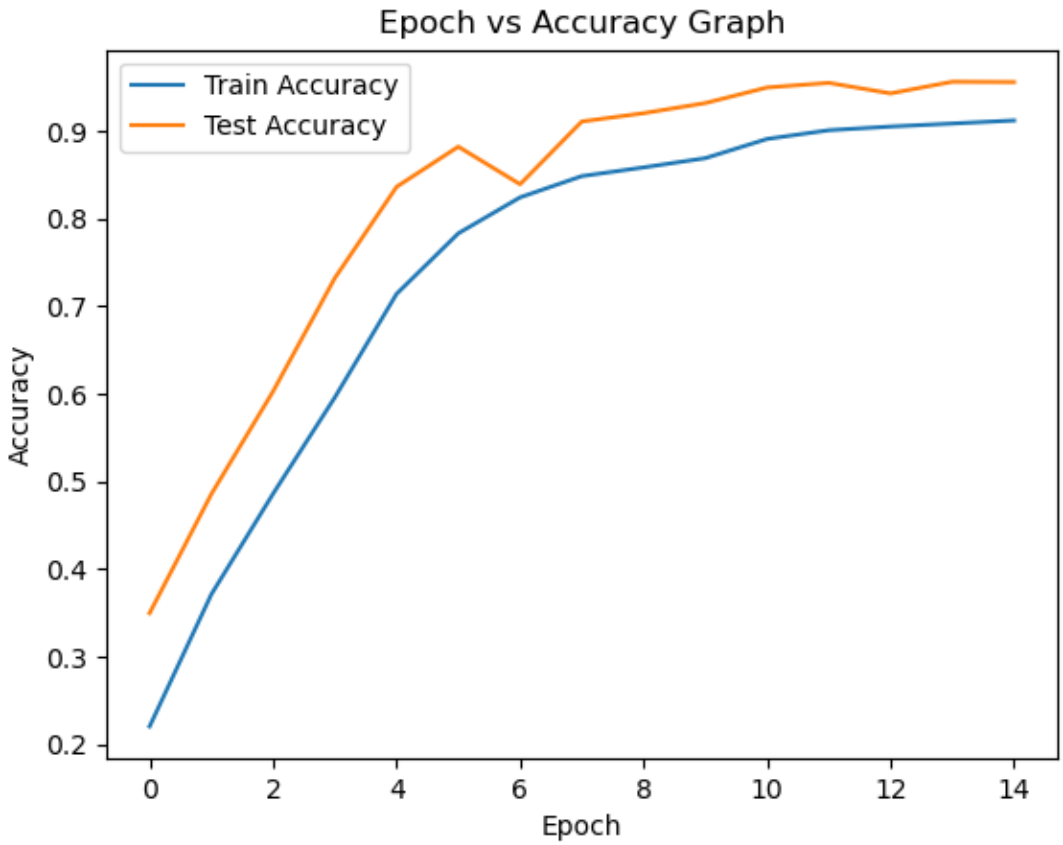

Figure 3: Epoch versus accuracy graph for epoch = 15, learning rate = 0.001, batch size = 64.

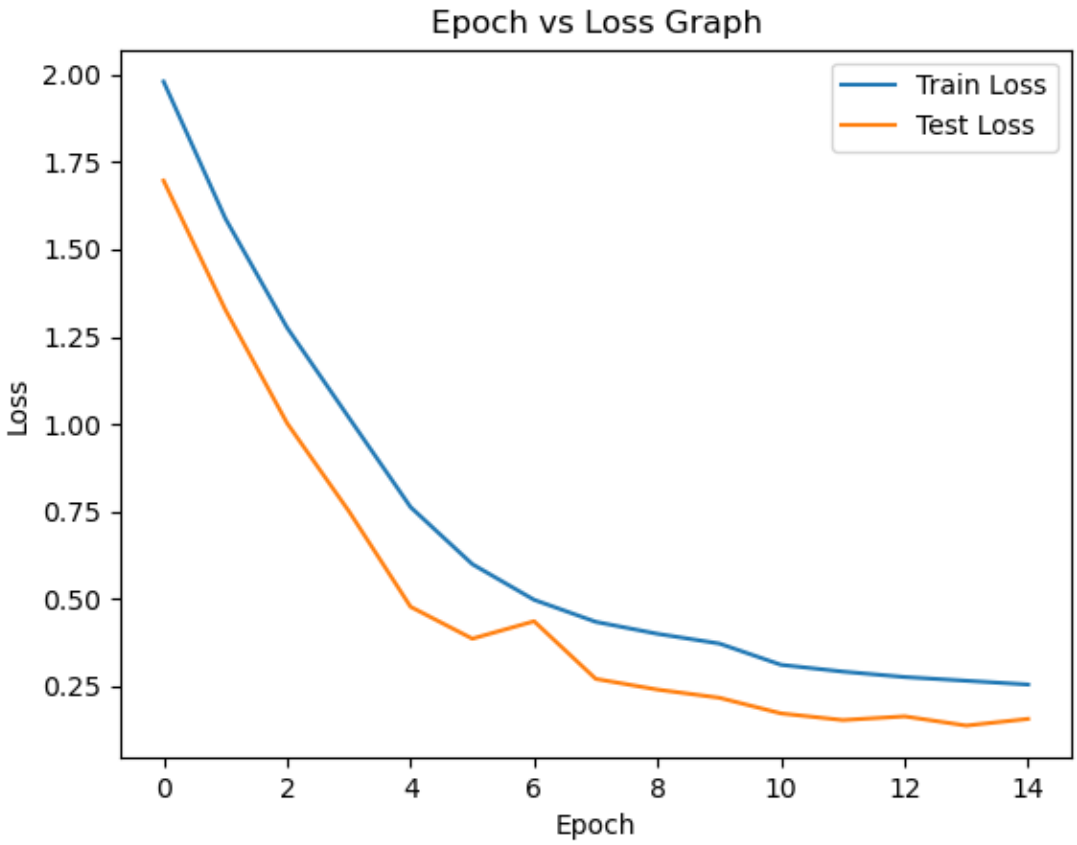

Figure 4: Epoch versus loss graph for epoch = 15, learning rate = 0.001, batch size = 64.

The figures given below displays the accuracy and loss results for 25 epochs, with a learning rate of 0.001 and 128 batch size:

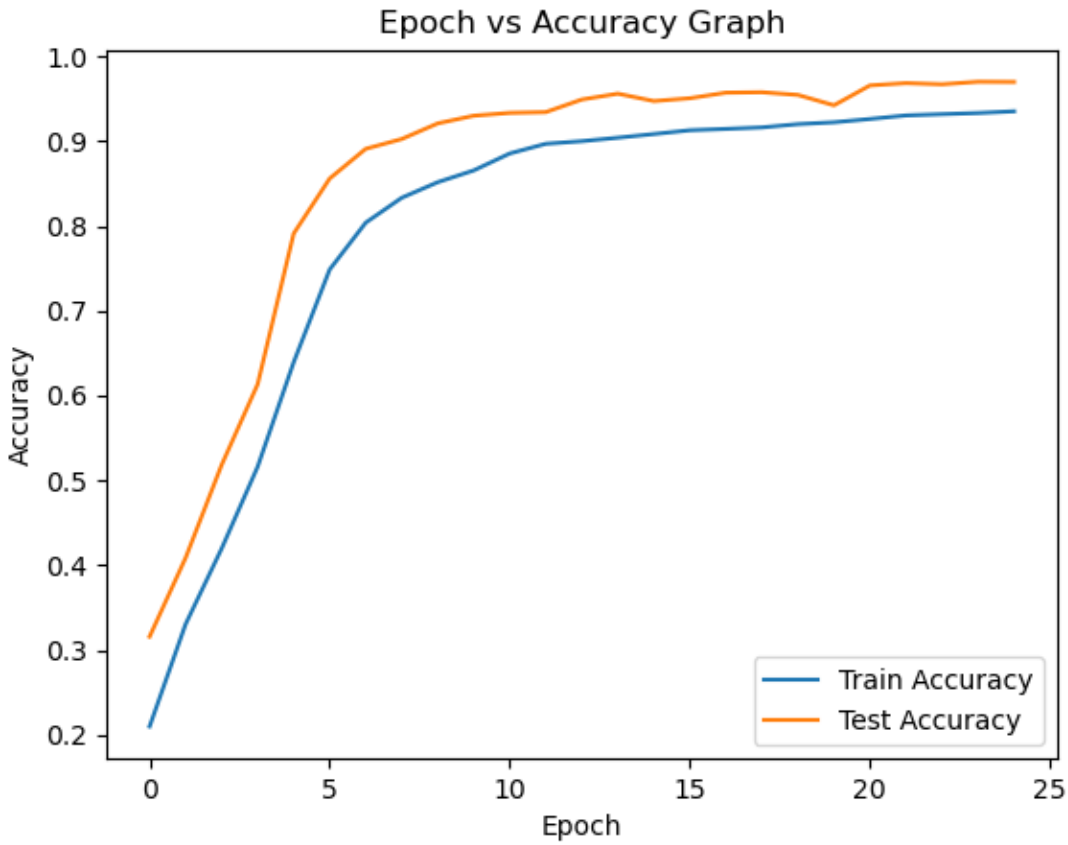

Figure 5: Epoch versus accuracy graph for epoch = 25, learning rate = 0.001, batch size = 128.

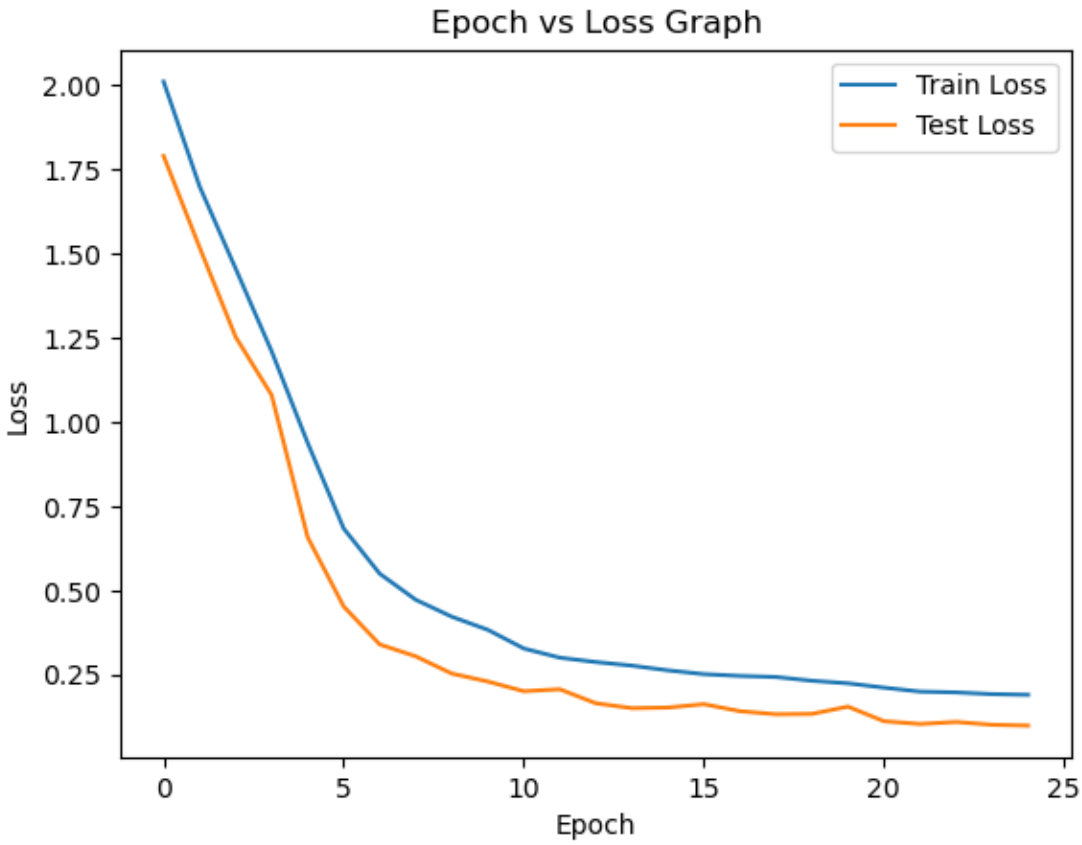

Figure 6: Epoch versus loss graph for epoch = 25, learning rate = 0.001, batch size = 128.

In all these test runs, we have accomplished an accuracy greater than 0.9 on both training and test sets. Hence, it can be said that both the architecture and parameters are satisfactory for the classification of the geometric shapes in given images.

An important thing that is noticed during the tests is that when we increased the learning rate to 0.005 or 0.01, the algorithm did not converge as we desired and stuck on an accuracy near 0.11. This might be occurring due to learning rate being too high, causing the optimizer to overshoot the optimal values during training. Hence, the model could not learn effectively.

Conclusion

In the sixth project of the course, a convolutional neural network was designed and implemented to classify geometric shapes in grayscale images with high accuracy. The model achieved strong performance with a learning rate of 0.001 and a batch size of 64, showing consistent convergence across different training runs. It was also observed that excessively high learning rates, such as 0.005 or 0.01, led to failure in convergence, with accuracy stuck around 0.11—showing signs of random performance.