

FAST NATIONAL UNIVERSITY CFD CAMPUS

|  |  |
| --- | --- |
| **Name:** | **Mariam Fatima**  **Muskan Ghani** |
| **Roll No:** | **22F-3168**  **22F-3841** |
| **Assignment:** | **6** |
| **Instructor:** | **Dr. Hashim Yaseen** |
| **Course:** | **Machine Learning** |

Comparison of CNN Architectures for MNIST Classification

# Introduction

This report evaluates the performance of three Convolutional Neural Network (CNN) architectures—LeNet-5, VGG-16, and a previously implemented custom CNN—on the MNIST dataset. The objective is to compare their test accuracies and computational efficiency to identify the most suitable model for this task.

# Architectures Evaluated

## 1. LeNet-5

Description:

One of the first CNN architectures, designed for digit recognition. It uses two convolutional layers followed by subsampling (pooling) layers, fully connected layers, and a softmax output layer.

Strengths: Lightweight, computationally efficient.

Weaknesses: Lacks the capacity to learn very complex features due to fewer layers and filters.

## 2. VGG-16

Description:

A deeper CNN with 16 layers, using 3×3 convolutional filters consistently and multiple pooling layers.

Strengths: Excellent at extracting hierarchical features due to its depth.

Weaknesses: Computationally expensive in terms of training time and memory requirements.

## 3. Custom CNN

Description:

Consists of two convolutional layers with 32 and 64 filters, followed by pooling and dropout layers, fully connected layers, and a softmax output layer.

Strengths: Balanced performance; specifically tailored for MNIST.

Weaknesses: Limited scope for scalability to more complex datasets.

# Experimental Setup

Dataset: MNIST (60,000 training images and 10,000 test images, each of size 28×28×1).

Preprocessing: Pixel values normalized to [0, 1].

## Hyperparameters:

Epochs: 10

Batch size: 128

Optimizer: Adam

Loss Function: Categorical Crossentropy

Environment: TensorFlow/Keras for implementation.

# Results

|  |  |  |
| --- | --- | --- |
| Model | Test Accuracy | Training Time per Epoch |
| LeNet-5 | 98.86% | ~6–9 seconds |
| VGG-16 | 99.20% | ~450–550 seconds |
| Custom CNN | 99.25% | ~11–14 seconds |

# Discussion

## 1. LeNet-5 Performance

LeNet-5 achieved a respectable test accuracy of 98.82%, making it a strong candidate for resource-constrained environments. However, its comparatively lower accuracy highlights its limited capacity for feature extraction due to the simplicity of its architecture.

## 2. VGG-16 Performance

VGG-16 demonstrated a high test accuracy of 99.20%, which is close to the custom CNN’s performance. The hierarchical feature extraction through its deep layers makes it highly effective. However, its significant computational cost (training time per epoch: ~450 seconds) makes it less practical for scenarios where computational efficiency is critical.

## 3. Custom CNN Performance

The custom CNN achieved the highest test accuracy of 99.25%, outperforming both LeNet-5 and VGG-16. The architecture is well-suited to MNIST due to its moderate depth and efficient training. It strikes a balance between accuracy and computational efficiency, requiring significantly less time per epoch compared to VGG-16.

# Conclusion

While all models performed well on MNIST, the custom CNN proved to be the most effective solution due to its balance of accuracy and computational efficiency. VGG-16 is recommended for applications requiring deeper insights but demands significantly more computational resources, whereas LeNet-5 serves as a lightweight alternative.