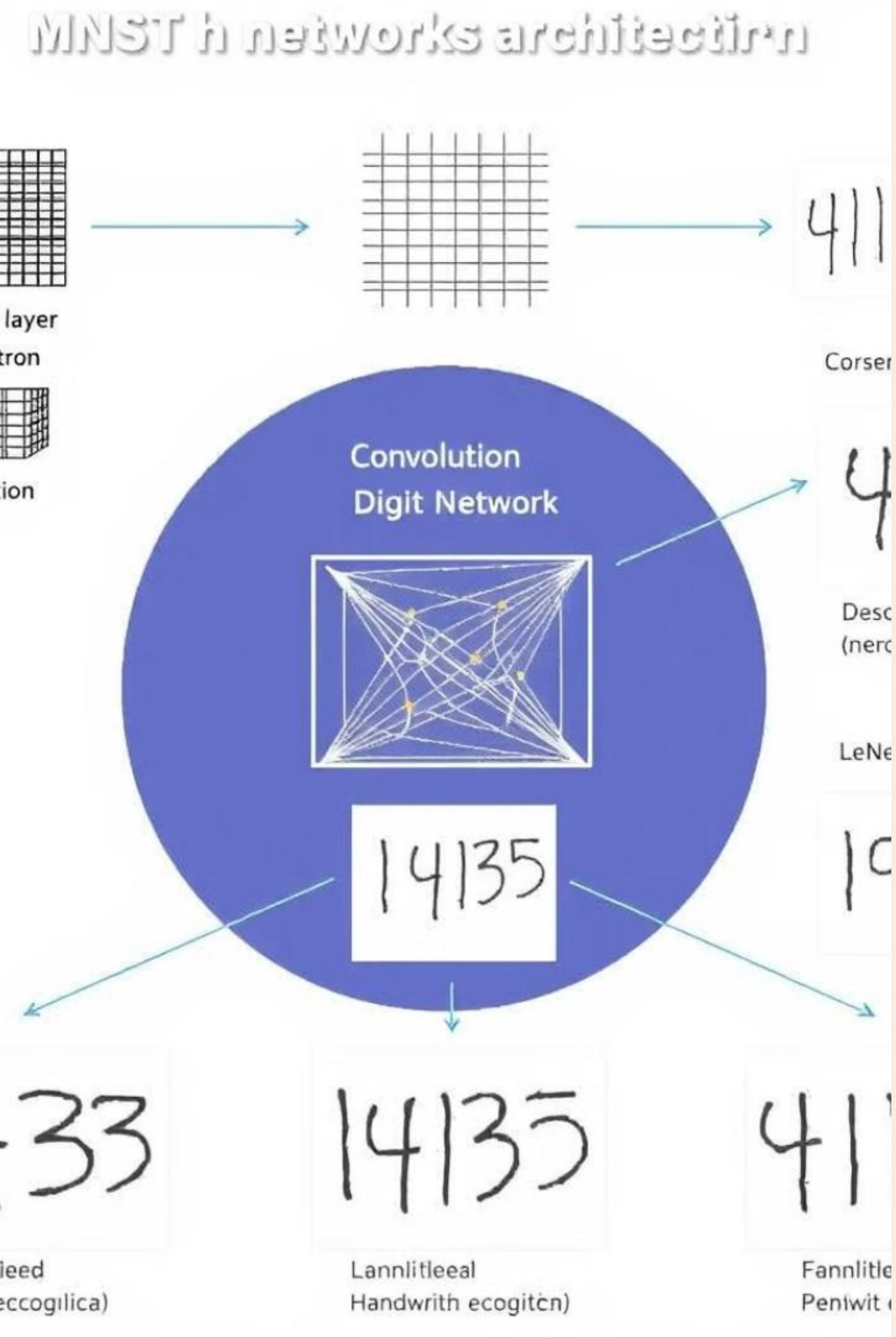


Handwritten Digit Recognition Using Neural Networks

This presentation explores the implementation of various neural network architectures for handwritten digit recognition using the MNIST dataset. We delve into the strengths and limitations of these models, including a Multilayer Perceptron (MLP), a Simple Convolutional Neural Network (CNN), and the renowned LeNet5 architecture.

Abhishek Kumar

Aim and Problem Statement

Objective

To develop and evaluate machine learning models capable of recognizing handwritten digits with high accuracy using the MNIST dataset.

Significance

This task is crucial in applications such as automated document processing, banking systems, and postal sorting.

Approach

We explored various neural network architectures, including a baseline MLP, a Simple CNN, and the LeNet5 model, to understand their strengths and limitations in digit recognition.

The MNIST Dataset

1 Description

The MNIST dataset consists of 60,000 training and 10,000 testing grayscale images of handwritten digits (0-9). Each image is 28x28 pixels, with corresponding labels representing the digits.

2 Significance

It is a standard benchmark dataset widely used to evaluate machine learning and deep learning algorithms.

3 Key Statistics

Image size: 28x28 pixels. Number of classes: 10. Training images: 60,000. Testing images: 10,000.

Preprocessing and Visualization

Preprocessing Techniques

1. Normalization: Scaled pixel values to the range [0, 1] for uniformity.
2. Conversion of images to tensors using the ToTensor transformation, enabling them to be used in PyTorch models.

Visualization

Displayed training sample dataset images to confirm preprocessing and data integrity.

Baseline Model - MLP

1

Architecture

Input Layer: 784 neurons. Hidden Layers: 512, 256 and 128 neurons, with ReLU activation. Output Layer: 10 neurons with Softmax activation.

2

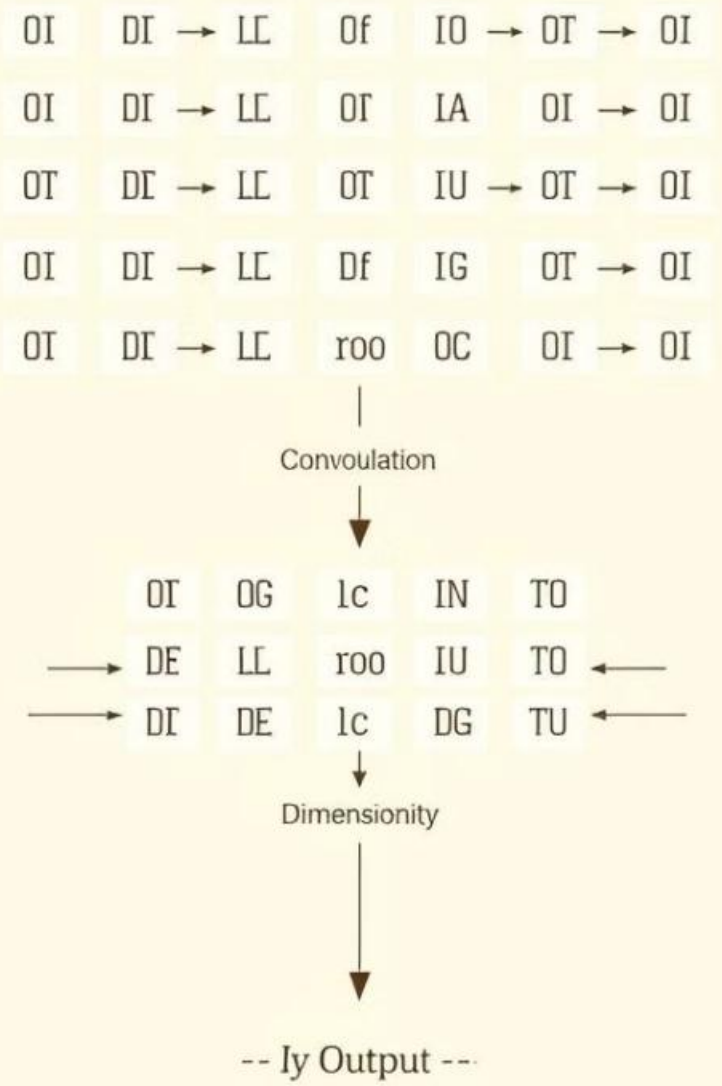
Hyperparameters

Optimizer: Adam. Loss Function: CrossEntropyLoss. Metrics: Accuracy, Precision, F1-Score.

3

Results

Accuracy: ~97.2%. Precision and F1-Score: High, with minor misclassifications in similar-looking digits.



Simple CNN Model

1

Architecture

Two convolutional layers with 32 and 64 filters, followed by MaxPooling layers. Fully connected layers: 128 neurons and an output layer with 10 neurons.

2

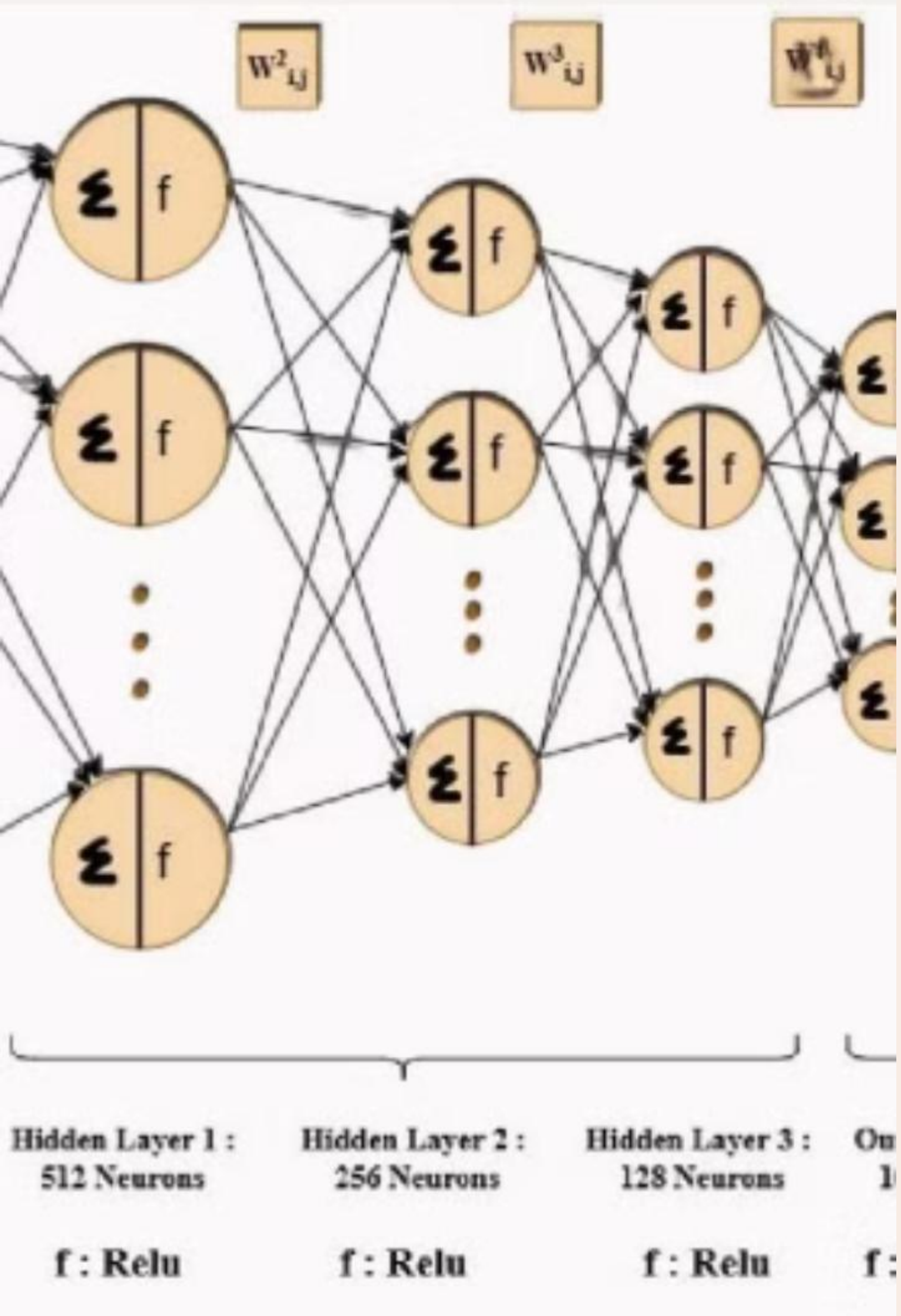
Hyperparameters

Optimizer: Adam. Loss Function: CrossEntropyLoss.

3

Results

Accuracy: ~99.08%. Precision and F1-Score: Improved consistency across all digit classes.



LeNet5 Model

Architecture	Two convolutional layers followed by dropout average pooling. Fully connected layers with layer leading to the output layer.
Hyperparameters	Optimizer: Adam. Learning Rate: 0.001.
Results	Accuracy: ~99.09% (highest among the models). Precision and F1-Score: Excellent performance across all classes.

Hyperparameter Tuning

Learning Rates

Explored various learning rates: 0.001, 0.0001, and 0.0005, to improve model performance.

Optimizers

Evaluated different optimizers: Adam, SGD, RMSprop.

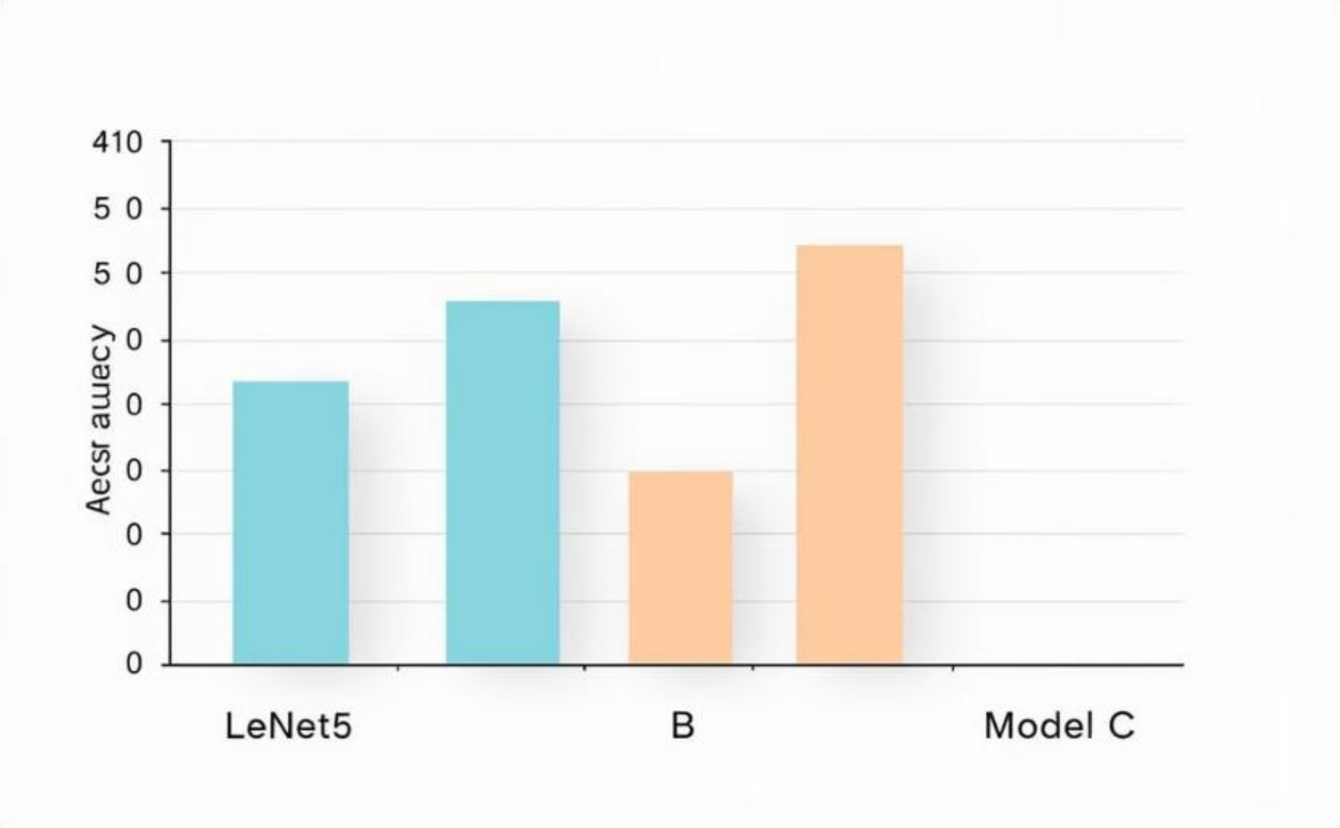
Kernel Sizes and Filters

Adjusted convolutional layer configurations.

Observations

LeNet5 performed best with Adam optimizer, a learning rate of 0.001 with learning rate scheduler, and carefully tuned filter sizes.

Conclusion



Model Performance

The project successfully implemented and evaluated three models for handwritten digit recognition. LeNet5 emerged as the most effective model with 99.09% accuracy.

Future Scope

Testing on more complex datasets and expanding the application to multi-digit recognition.