INFOSYS SPRINGBOARD

INTERNSHIP REPORT

HANDWRITTEN DIGIT RECOGNITION MODEL

Project Mentor:

Narendra Kumar

Presented by:

Group 1

Akhil Jain

Sachin Rajawat

Pranav Gupta

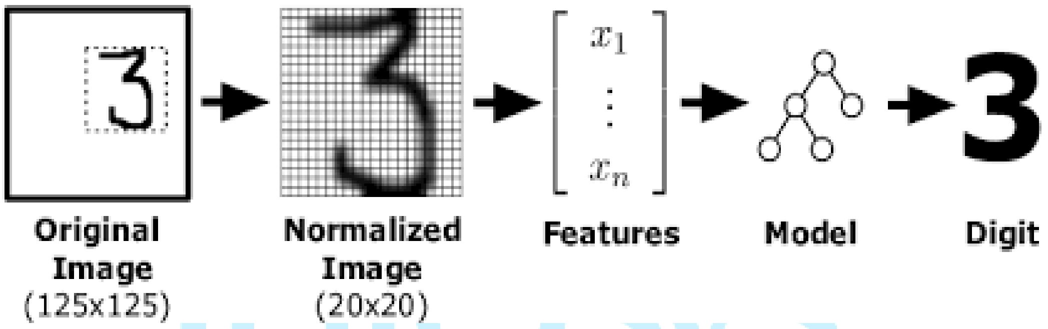
Sarang Kishor Masurkar

Shahnaz Ali

Table of Contents

• INTRODUCTION:

A popular demonstration of the capability of deep learning techniques is object recognition in image data. One of the most fundamental projects of object recognition for machine learning and deep learning is the MNIST dataset for handwritten digit recognition. Handwritten digit recognition is a multiclass supervised learning problem. In our work, we have additionally used various variants of the EMNIST dataset , which has more labels and data points, that aims to add a level of complexity to the project.



• DATASET OVERVIEW:

MNIST Dataset

- **Purpose:** Benchmark for handwritten digit classification.
- **Content:** 70,000 grayscale images (60,000 training and 10,000 testing) of digits (0-9), each 28x28 pixels.
- **Usage:** Widely used for developing and testing image classification algorithms.

EMNIST Dataset (Extended MNIST)

- **Purpose:** Extends MNIST to include handwritten letters and digits.
- **Content:** Over 800,000 grayscale images of digits (0-9), uppercase (A-Z), and lowercase letters (a-z), each 28x28 pixels.
- **Variants:** Includes subsets like balanced, byclass, bymerge, digits, and letters.
- **Usage:** Suitable for more complex classification tasks involving alphanumeric characters.

• ALGORITHMS USED:

1. MLP Model:

The model is a simple neural network with one hidden layer with the same number of neurons as there are inputs (784). A rectifier activation function is used for the neurons in the hidden layer. The output of this model are **logits**, meaning they are real numbers which can be transformed into probability-like values using a ReLU function.

Architecture and Layers

- **Fully Connected Layers**:
  - fc1: Linear(784, 128)
  - fc2: Linear(128, 64)
  - fc3: Linear(64, 10)

Activation Function

- **ReLU**: The ReLU activation function introduces non-linearity, allowing the model to learn complex patterns.

Normalization

- **Standardization**: Normalization with mean 0.5 and standard deviation 0.5 helps in stabilizing and speeding up the training process.

Optimizer

- **Adam**: Adaptive moment estimation optimizer that combines the advantages of two other extensions of stochastic gradient descent. It computes individual adaptive learning rates for different parameters.

Input Size

- **28x28**: The input size is a flattened 28x28 image, which is 784 pixels.

Number of Parameters

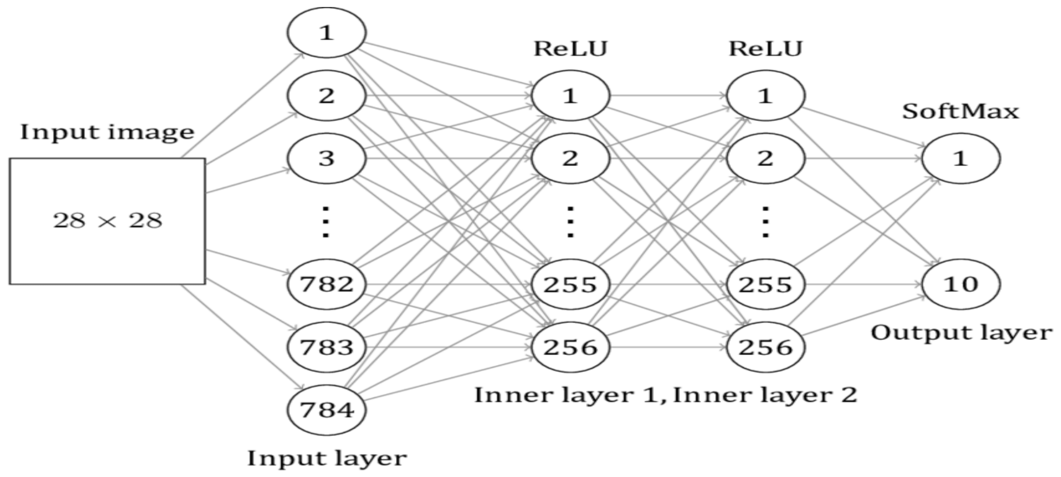
- **109,386**: This model has a relatively high number of parameters due to the fully connected layers, which can make it prone to overfitting if not properly regularized.

Loss Function

- **CrossEntropyLoss**: Suitable for classification tasks, it measures the performance of a classification model whose output is a probability value between 0 and 1.

Analysis:

- **Strengths:**
  - Simplicity: The MLP model is straightforward, making it easier to implement and train.
  - Fully connected layers can capture global patterns.
  - The model is lightweight compared to more complex architectures.
- **Weaknesses:**
  - Lack of spatial awareness: Since MLP flattens the input, it may lose spatial information, which is crucial for image data.
  - Higher number of parameters compared to CNN models, leading to potential overfitting if not managed properly.
- **Performance:**
  - The MLP model might show reasonable performance on simple datasets but can struggle with more complex image recognition tasks.
  - It can be computationally intensive due to the fully connected nature, especially as input dimensions increase.



2. Simple CNN Model:

A convolutional neural network that has more layers than the baseline MLP was employed, which was successfully able to increase the accuracy . This model consists of convolution, pooling , fully connected layers in addition to Relu for activation.

Architecture and Layers

- **Convolutional Layers:**
  - conv1: Conv2d(1, 16, 5)
  - conv2: Conv2d(16, 32, 5)
- **Max Pooling Layers:**
  - After conv1 and conv2: MaxPool2d(kernel_size=2)
- **Fully Connected Layers:**
  - fc1: Linear(512, 128)
  - fc2: Linear(128, 10)

Activation Function

- **ReLU:** The ReLU activation function introduces non-linearity, allowing the model to learn complex patterns.

Normalization

- **Standardization:** Normalization with mean 0.5 and standard deviation 0.5 helps in stabilizing and speeding up the training process.

Optimizer

- **Adam:** Adaptive moment estimation optimizer that combines the advantages of two other extensions of stochastic gradient descent. It computes individual adaptive learning rates for different parameters.

Input Size

- **28x28:** The input size is a 28x28 image.

Number of Parameters

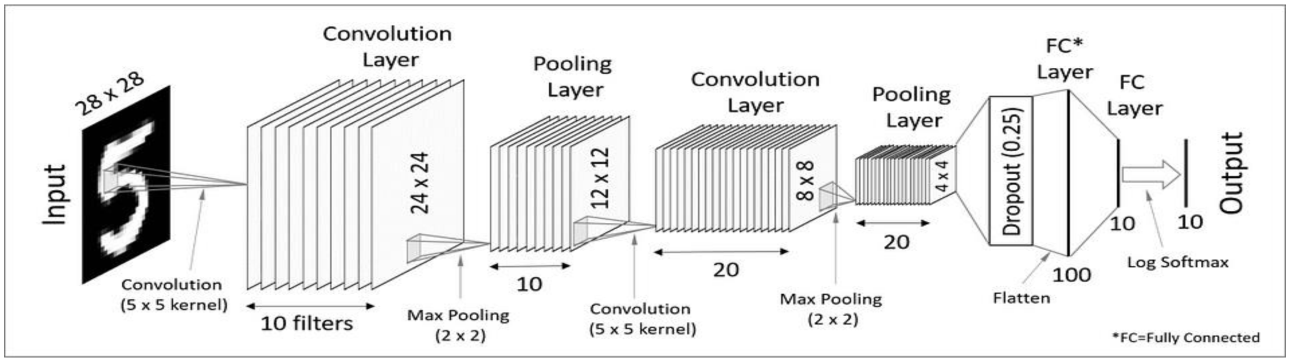
- **80,202:** This model has fewer parameters compared to the MLP model due to its convolutional nature.

Loss Function

- **CrossEntropyLoss:** Suitable for classification tasks, it measures the performance of a classification model whose output is a probability value between 0 and 1.

Analysis:

- **Strengths:**
  - **Spatial Pattern Recognition:** CNNs are well-suited for image data as they can capture spatial hierarchies and learn feature hierarchies automatically.
  - **Efficient Parameter Use:** The model efficiently reduces parameters through convolutional layers.
- **Weaknesses:**
  - **Complexity:** Requires careful tuning of convolutional layers and may be more computationally intensive than MLPs due to larger parameter sizes.
- **Performance:**
  - **Better for Image Data:** The Simple CNN model generally performs better on image recognition tasks compared to MLP due to its ability to capture spatial features.
  - **Computational Needs:** While computationally more efficient than fully connected networks, it still requires significant resources, especially for larger datasets.



3. LeNet 5 Model:

One of the earliest demonstrations of the effectiveness of convolutional layers in neural networks is the “LeNet5” model. This model is developed to solve the MNIST classification problem. It has three convolutional layers and two fully connected layers to make up five trainable layers in the model.

Architecture and Layers

- **Convolutional Layers:**
  - conv1: Conv2d(1, 6, 5)
  - conv2: Conv2d(6, 16, 5)
- **Max Pooling Layers:**
  - After conv1 and conv2: MaxPool2d(kernel_size=2)
- **Fully Connected Layers:**
  - fc1: Linear(256, 120)
  - fc2: Linear(120, 84)
  - fc3: Linear(84, 10)

Activation Function

- **ReLU:** The ReLU activation function introduces non-linearity, allowing the model to learn complex patterns.

Normalization

- **Standardization:** Normalization with mean 0.5 and standard deviation 0.5 helps in stabilizing and speeding up the training process.

Optimizer

- **Adam:** Adaptive moment estimation optimizer that combines the advantages of two other extensions of stochastic gradient descent. It computes individual adaptive learning rates for different parameters.

Input Size

- **28x28:** The input size is a 28x28 image.

Number of Parameters

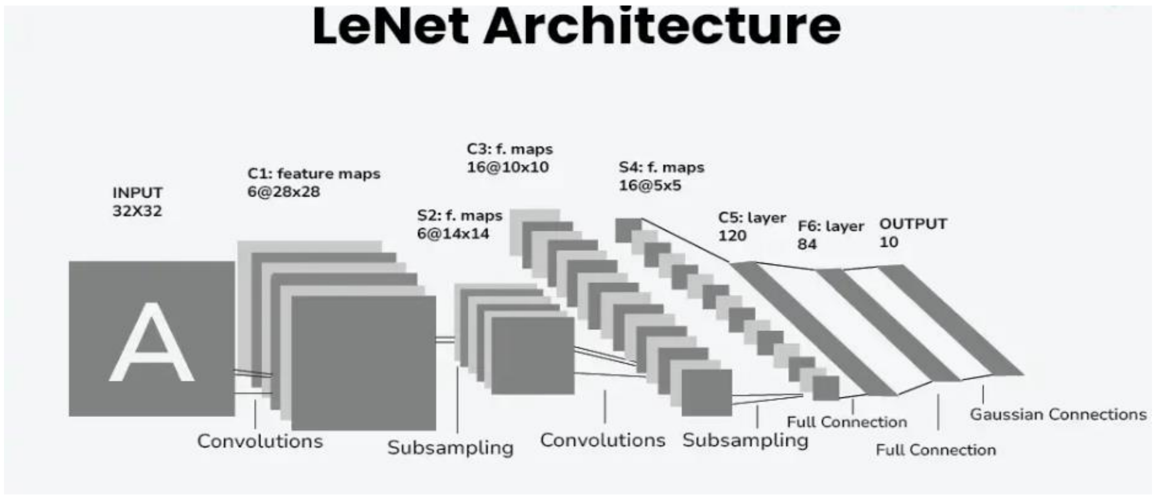
- **44,186:** This model has the fewest parameters among the three models due to its efficient design.

Loss Function

- **CrossEntropyLoss:** Suitable for classification tasks, it measures the performance of a classification model whose output is a probability value between 0 and 1.

Analysis:

- **Strengths:**
  - **Classic and Proven:** LeNet-5 is a classic CNN architecture known for its effectiveness in handwritten digit recognition tasks.
  - **Efficiency:** It strikes a balance between model complexity and efficiency, making it less computationally intensive.
- **Weaknesses:**
  - **Older Design:** May not capture very fine details in complex images due to its older architecture design.
- **Performance:**
  - **Balanced Performance:** LeNet-5 generally offers balanced performance, making it suitable for a wide range of image recognition tasks without being too resource-heavy.
  - **Computationally Efficient:** Less computationally intensive than more modern and complex architectures, making it suitable for deployment on less powerful hardware.



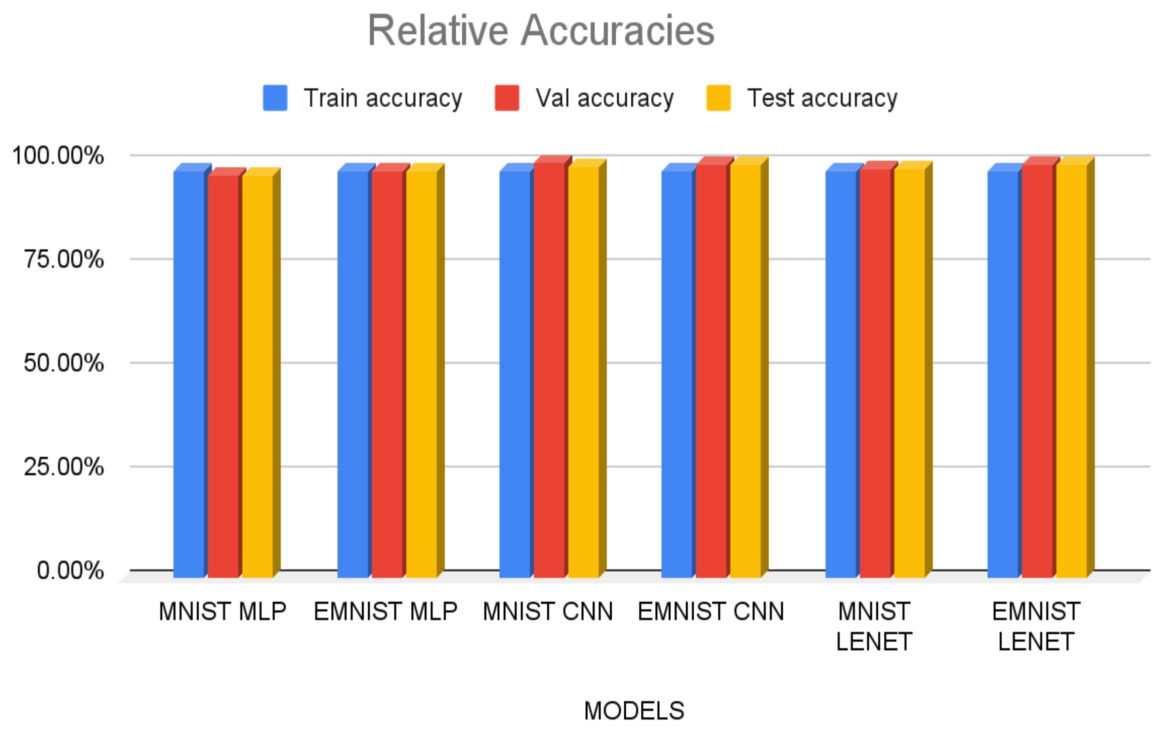
• COMPARISON BETWEEN MODELS:

On experimenting with different datasets , specified models and tweaking various variables, we have tried to sum up our conclusions in the following table. Every model , takes an input image of 28 x 28 pixels.

Features	Inputs					
Model	MLP		Simple CNN		LeNet 5	
Architecture	Fully Connected		Convolutional		Convolutional	
Dataset Used	MNIST	EMNIST	MNIST	EMNIST	MNIST	EMNIST
Data Description	10 labels, 70,000 data points	10 labels, 280,000 data points	10 labels, 70,000 data points	10 labels, 280,000 data points	10 labels, 70,000 data points	10 labels, 280,000 data points
Type of Model	Simple Neural Network		Convolutional Neural network		Deep Neural Network	
Layers	3		6		7	
Epochs	10		10		10	
Input Shape	28x28		28x28		28x28	
Activation Function	ReLU		ReLU		ReLU	
Optimizer	Adam		Adam		Adam	
Loss Function	CrossEntropyLoss		CrossEntropyLoss		CrossEntropyLoss	
Normalization	Yes ((0.5,), (0.5,))		Yes ((0.5,), (0.5,))		Yes ((0.5,), (0.5,))	
No. of Parameters	109,386		80,202		44,186	
Training Accuracy	97.94%	98.97%	99.65%	99.79%	99.42%	99.62%
Validation Accuracy	96.88%	98.28%	98.98%	99.43%	98.84%	99.37%
Testing Accuracy	96.87%	98.28%	99.09%	99.41%	98.77%	99.39%

• OBSERVATIONS:

CNN and Lenet models outperformed the baseline MLP, across all datasets that were used.



As the complexity of the dataset increases, performance decreases. So, we chose MNIST trained models for our final UI as they performed better.