Handwritten Digit Recognition with Convolutional Neural Network

Sudipta Ghosh
Roll No.: 23MA60R03

Supervisor: Prof. Adrijit Goswami

Indian Institute of Technology Kharagpur
West Bengal-721302, India

Handwritten Digit Recognition: A Timeless Challenge

- **What is Handwritten Digit Recognition?**

  Handwritten digit recognition(HCR) is a critical task in computer vision focused on the automated identification of handwritten numerals.

- **Applications in Real Life:**
  - **Postal Systems:** Automating postal code recognition for efficient mail sorting.
  - **Banking:** Processing handwritten checks and forms.
  - **Historical Research:** Digitizing historical manuscripts and archives.
  - **Scientific Research:** Streamlining data collection from handwritten experiments.

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Objective

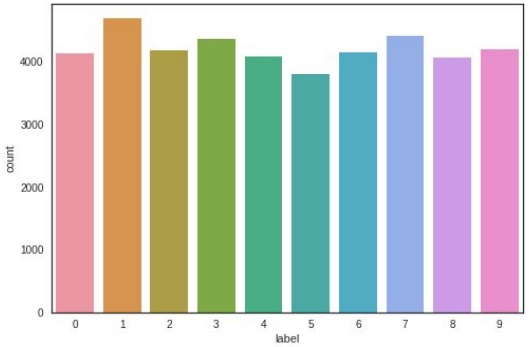
- Dataset Description
- Data Preprocessing
- Build a CNN model to recognize handwritten digits.
- Performance Analysis
- Challenges and Solutions
- Comparison with other Machine Learning Models
- Deployment

Dataset Description

- **Dataset Composition:**
  - 60,000 training images and 10,000 testing images.
  - Each image represents a handwritten digit (0 to 9).
- **Image Specifications:**
  - Grayscale images with a fixed size of 28 x 28 pixels.
  - Can be flattened into a 1D array of 784 values (28 x 28).
- **Storage Format:**
  - Images stored in one file in a random order.
  - Each image has a corresponding label (digit it represents).
- **Contributors:**
  - Written by 250 people:
  - Half are Census Bureau employees. Half are students.

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Visualization



Digit count in Dataset

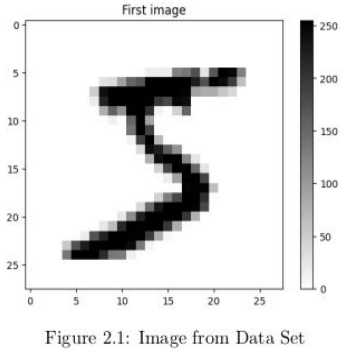
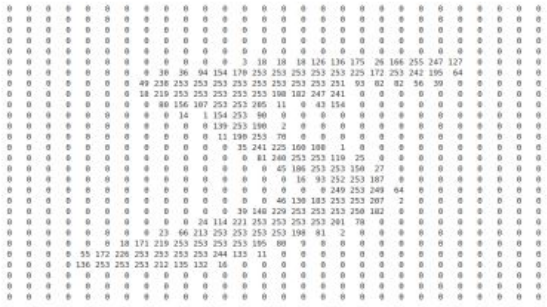


Figure 2.1: Image from Data Set

image from Dataset



Pixel representation of image

Data Preprocessing

- **Normalization:** We perform a grayscale normalization to reduce the effect of illumination's differences. Moreover the CNN converge faster on [0..1] data than on [0..255].

- **Reshape:** Train and test images (28px x 28px) has been stock into pandas.Dataframe as 1D vectors of 784 values. We reshape all data to 28x28x1 3D matrices. Keras requires an extra dimension in the end which correspond to channels. MNIST images are gray scaled so it use only one channel. For RGB images, there is 3 channels, we would have reshaped 784px vectors to 28x28x3 3D matrices.

- **Label Encoding:** Label are 10 digits numbers from 0 to 9. We need to encode these label to one hot vectors(ex : 2 -> [0,0,1,0,0,0,0,0,0,0]).

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Proposed Model: Convolutional Neural Networks(CNN)

Convolutional Neural Networks (CNNs) are a powerful class of deep learning models that have revolutionized the field of computer vision. They are specifically designed to process visual imagery, such as images, videos, and medical scans. CNNs excel at tasks like image classification, object detection, and image segmentation.

Key Components of a CNN:

- **Convolutional Layers:** Extract features from input images using filters.
- **Activation Functions:** Introduce non-linearity to the network.
- **Pooling Layers:** Reduce the spatial dimensions of feature maps.
- **Fully Connected Layers:** Classify the extracted features.

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Layers of CNN

- Convolution Layer
- Polling Layer
- Fully Connected Layer

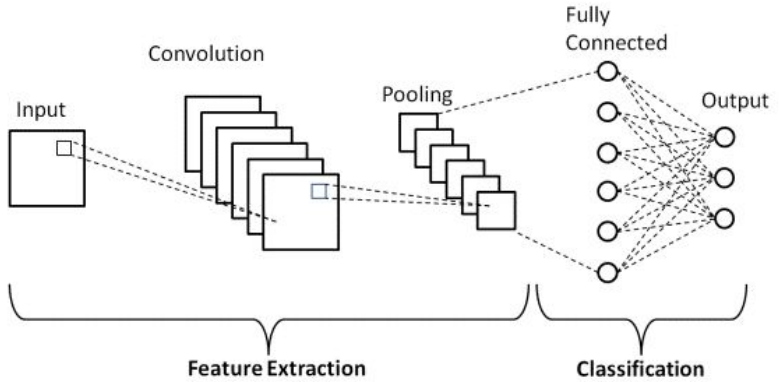


Fig: CNN model Architecture

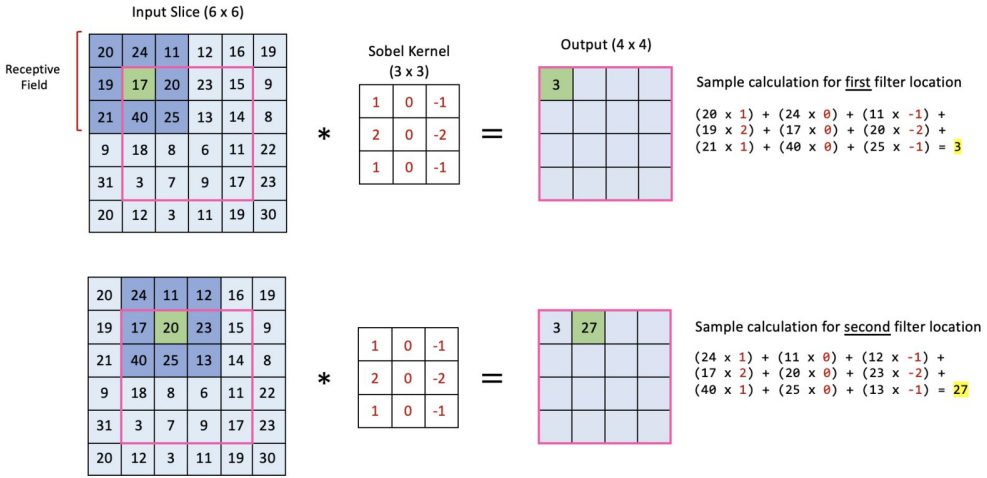
Input Layer:

- A 28x28 image of the digit ”7” is fed into the CNN.
- Each pixel has a value between 0 (black) and 255 (white).

Convolution Layer

Purpose: Extracts features such as edges, textures, and patterns.

How it Works: Applies a filter/kernel (e.g., 3x3 matrix) that slides over the image. Performs element-wise multiplication and sums the results to produce a feature map.



Convolution Layer (Contd...)

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
255	255	255	255	255	255
255	255	255	255	255	255
255	255	255	255	255	255

*

-1	-1	-1
0	0	0
1	1	1

Filter/Kernel

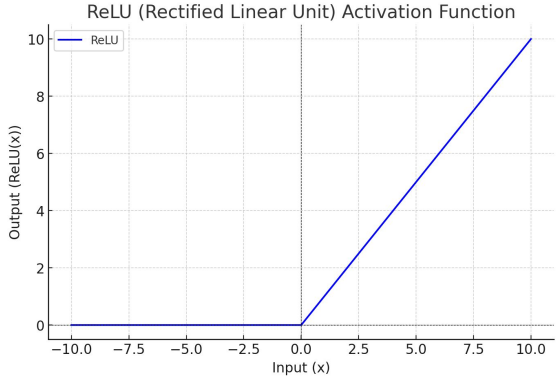
=

0	0	0	0
255	255	255	255
255	255	255	255
0	0	0	0

Activation Layer (ReLU)

Purpose: Introduces non-linearity into the model.

- Applies the ReLU function: **f(x) = max(0, x)**
- ReLU ensures that negative values in the feature map are set to 0.



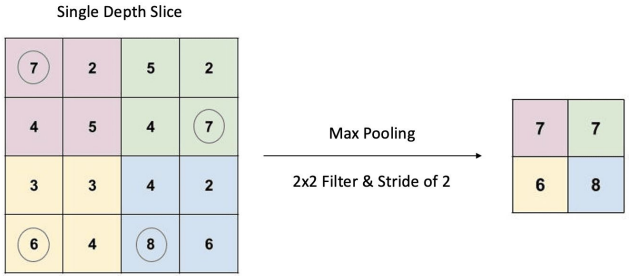
Note: After the convolution, any negative pixel values are converted to 0, emphasizing only important features.

Pooling Layer

Purpose: Reduces the spatial dimensions of feature maps while retaining key features.
Types: Max Pooling (most common) and Average Pooling.

How it Works:

- Applies a 2x2 window over the feature map.
- For Max Pooling, it keeps the maximum value in each window.



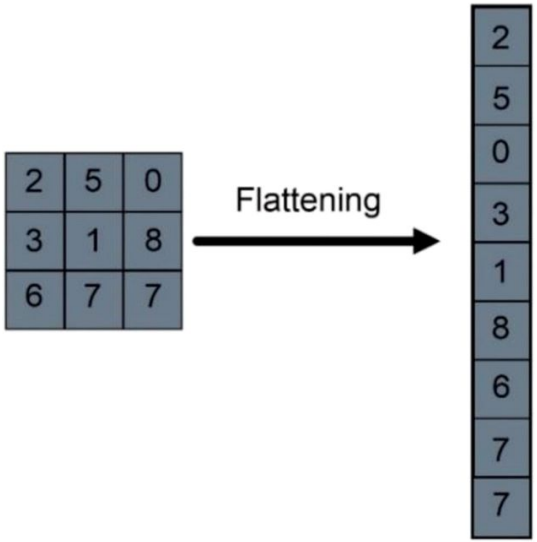
Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Flattening Layer

Purpose: Converts the 2D feature maps into a 1D vector.

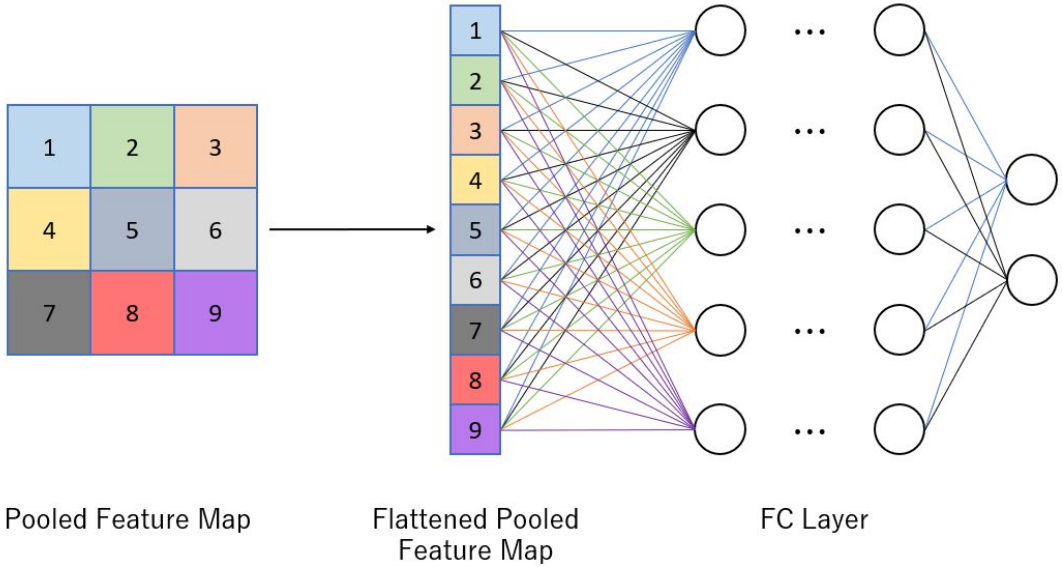
How it Works: The output of the pooling layer (e.g., 12x12 matrix) is flattened into a 1D array of size 144.

And that is the input of Dense Layer.



Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Flattening Layer → Dense Layer



Fully Connected (Dense) Layer

Purpose: Maps extracted features to output categories.

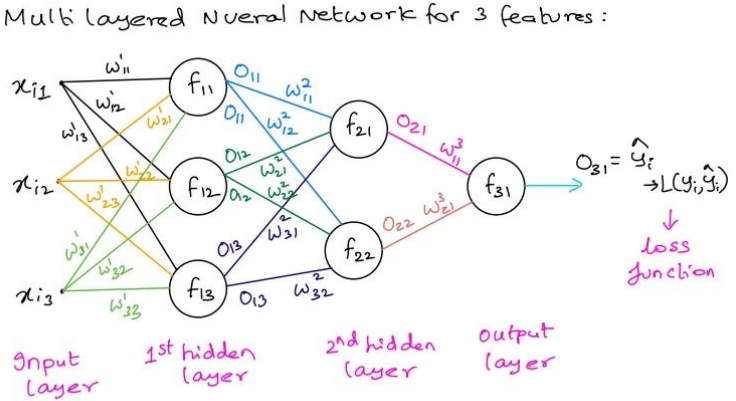
How it Works: Each neuron in this layer is connected to every value in the previous layer. Applies weights and biases to compute probabilities.

$$g(w_{11}^1 x_{i1} + w_{21}^1 x_{i2} + w_{31}^1 x_{i3}) = O_{11}$$

$$g(w_{12}^1 x_{i1} + w_{22}^1 x_{i2} + w_{32}^1 x_{i3}) = O_{12}$$

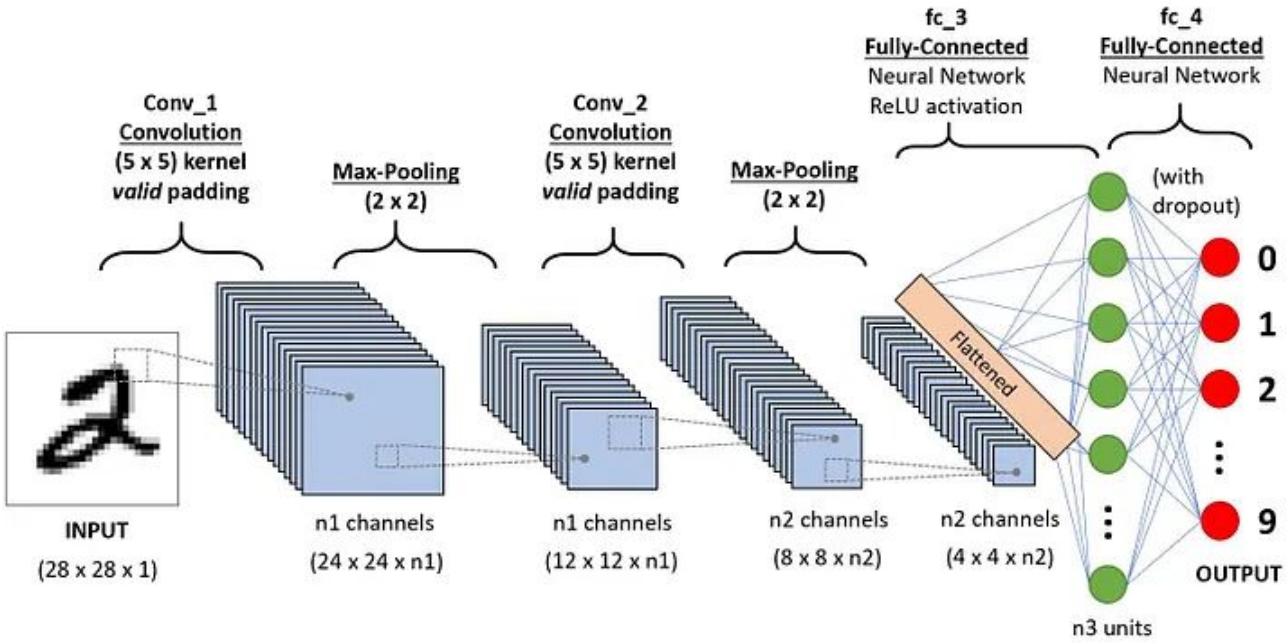
$$g(w_{13}^1 x_{i1} + w_{23}^1 x_{i2} + w_{33}^1 x_{i3}) = O_{13}$$

where g is **Activation function**.



Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Convolution Neural Network(CNN)



Proposed CNN Architecture

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 32)	832
conv2d_1 (Conv2D)	(None, 28, 28, 32)	25,632
max_pooling2d (MaxPooling2D)	(None, 14, 14, 32)	0
dropout (Dropout)	(None, 14, 14, 32)	0
conv2d_2 (Conv2D)	(None, 14, 14, 64)	18,496
conv2d_3 (Conv2D)	(None, 14, 14, 64)	36,928
max_pooling2d_1 (MaxPooling2D)	(None, 7, 7, 64)	0
dropout_1 (Dropout)	(None, 7, 7, 64)	0
flatten (Flatten)	(None, 3136)	0
dense (Dense)	(None, 256)	803,072
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 10)	2,570

Total params: 1,775,062 (6.77 MB)
Trainable params: 887,530 (3.39 MB)
Non-trainable params: 0 (0.00 B)
Optimizer params: 887,532 (3.39 MB)

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Loss Function, Optimizer, Analeator

Loss function: to measure how poorly our model performs on images with known labels. We use a specific form for categorical classifications (>2 classes) called the "categorical_crossentropy".

$$\text{Loss} = -\sum_{i=1}^{\text{output size}} y_i \cdot \log \hat{y}_i$$

Multiclass Cross-Entropy

Optimizer: This function will iteratively improve parameters (filters, kernel values, weights and bias of neurons ...) in order to minimise the loss. I have chosen RMSprop (with default values).

Analeator: In order to make the optimizer converge faster and closest to the global minimum of the loss function, i used an annealing method of the learning rate (LR). I have used the ReduceLROnPlateau function from Keras.

Model Performance Metrics

The model reaches almost **99% (98.7+%) accuracy** on the validation dataset after 15 epochs. The validation accuracy is greater than the training accuracy almost every time during the training. That means that our model dosen't not overfit the training set.

Our model is very well trained !!!

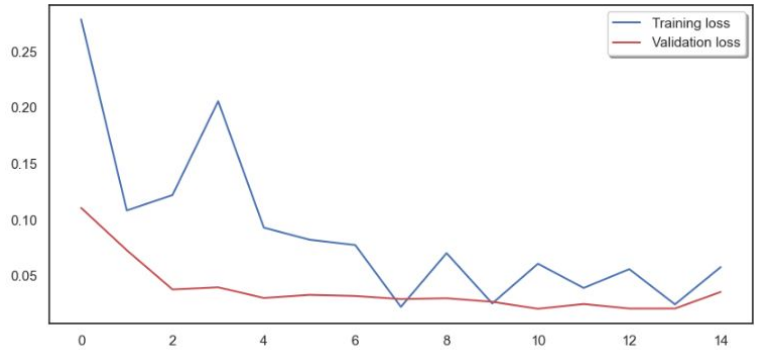


Fig: Training Loss vs Validation Loss

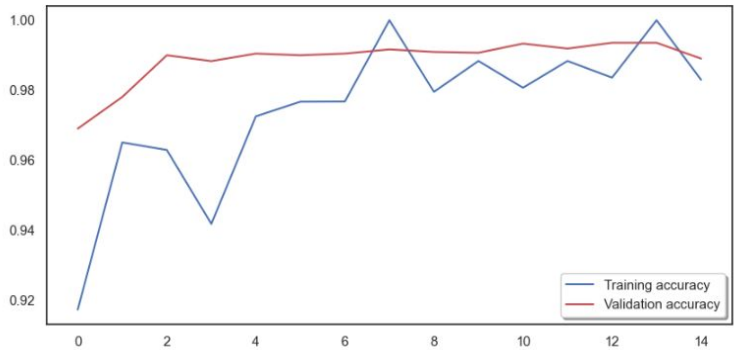


Fig: Training vs Validation accuracy

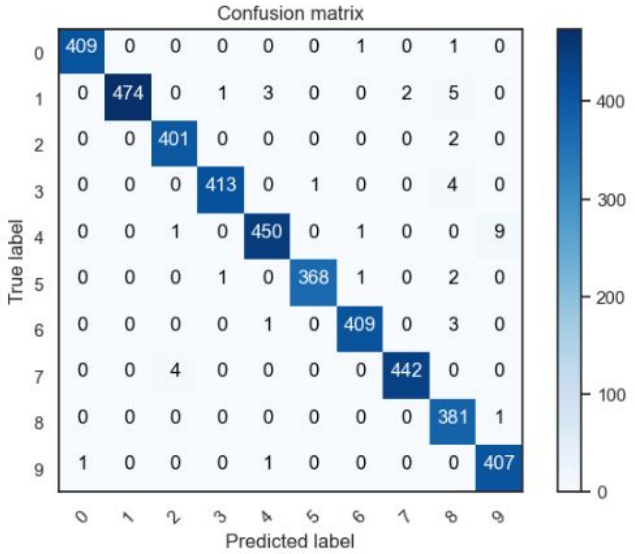
Confusion Matrix

Confusion matrix can be very helpful to see our model drawbacks.

Pression: 0.99

Recall: 0.99

F1 Score: 0.99

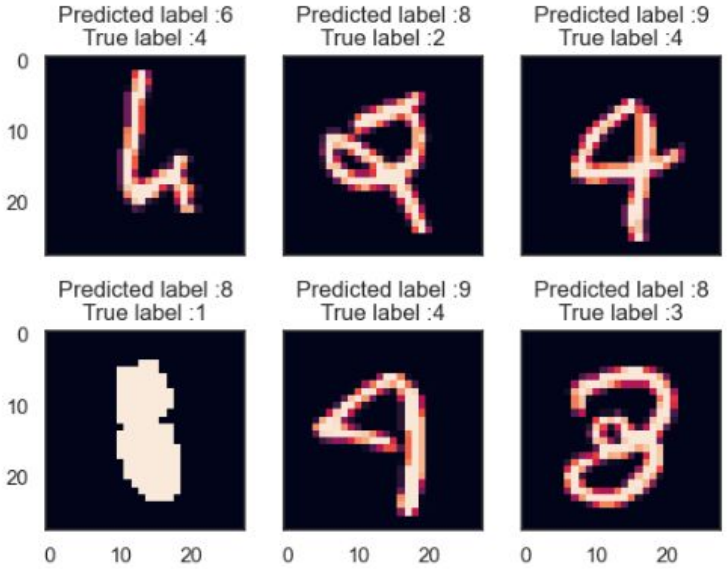


Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

Error Identification

- Here we can see that our CNN performs very well on all digits with few errors considering the size of the validation set (4200 images).

- Our CNN performs well on most digits, but struggles with **4s**, often misclassifying them as **9s**, especially when handwriting is smooth.



Challenges & Solutions

Challenges:

- Overfitting.
- Selection of hyperparameters.
- Computational resource constraints.

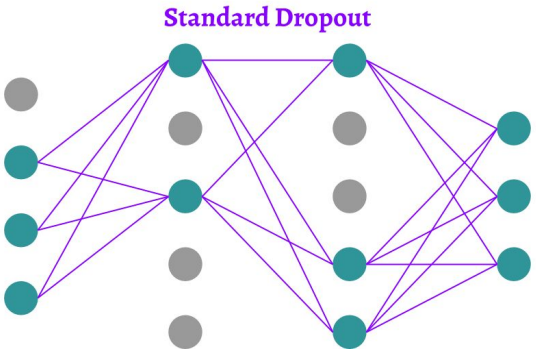
Solutions:

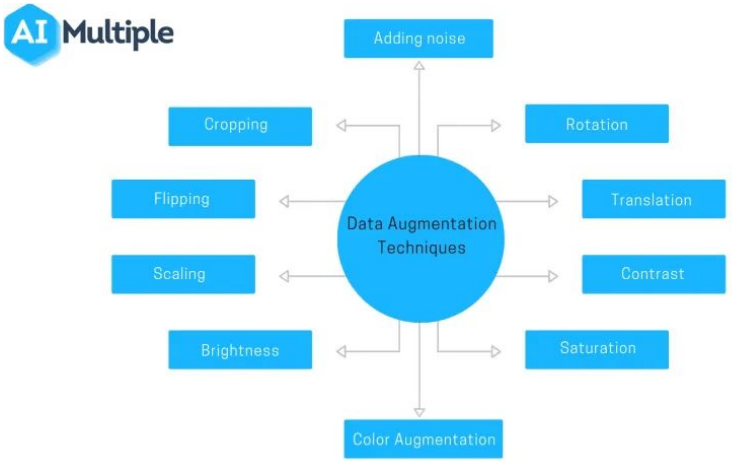
- Data augmentation.
- Dropout layers.
- Regularization techniques.
- Hyperparameter tuning.

Data Augmentation and Dropout

Dropout: a regularization technique in deep learning that involves randomly removing units and their connections from a neural network during training.

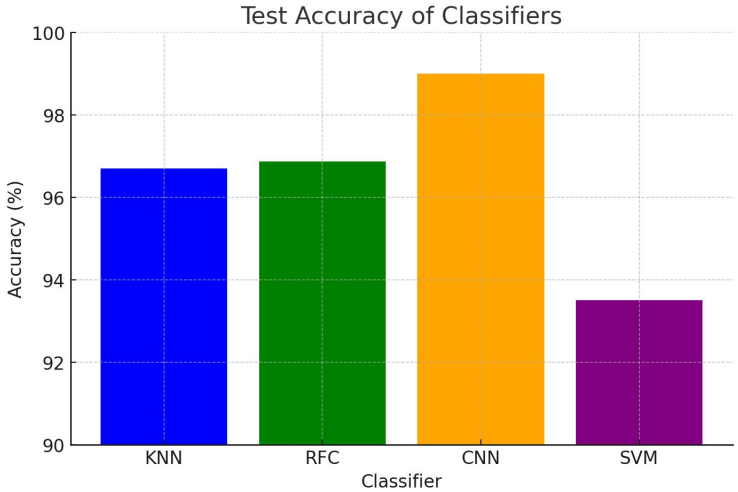
Data augmentation: is a technique that creates new data from existing data to improve Deep learning (DL) models.



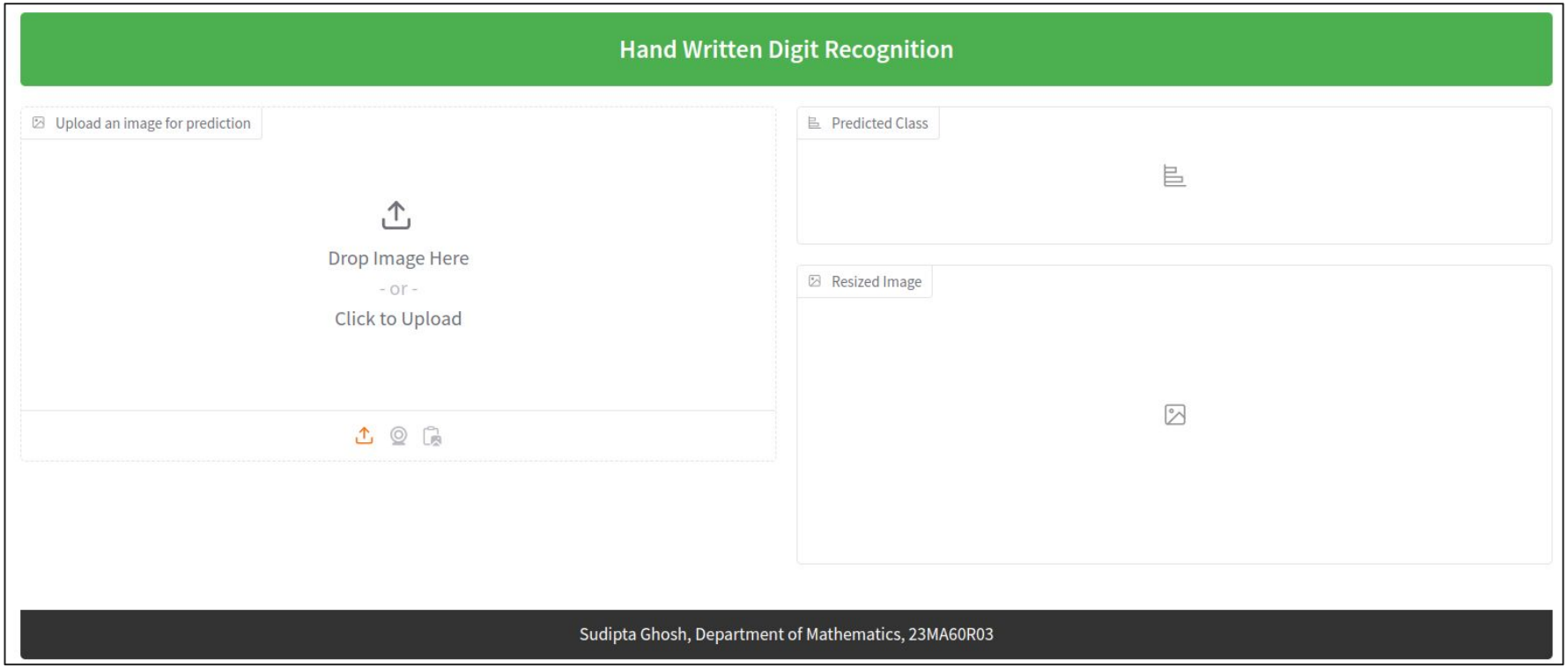


Comparison with Other Approaches

I compare the results of four classification algorithms namely **K-Nearest Neighbors(KNN)**, **Convolutional Neural Network(CNN)**,**Random Forest Classifier(RFC)** and **Support Vector Machine(SVM)** on the MNIST database.



Deployment (Web Interface)



Future Work

Document Analysis and OCR:

- Build a complete OCR system to digitize documents, detect handwriting, and convert to text.
- Extend the project to include printed text recognition alongside handwriting.

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur

References

[1] Xiaofeng Han and Yan Li (2015), “The Application of Convolution Neural Networks in Handwritten Numeral Recognition” in International Journal of Database Theory and Application, Vol. 8, No. 3, pp. 367-376.

[2] L. Bottou, C. Cortes, “Comparison of Classifier methods a case study in handwritten digit recognition”, Pattern Recognition, 1994. Vol. 2 Conference B; Image Processing, Proceedings of the 12th IAPR International. Conference IEEE, 06 August 2002.

[3] Seiichi Uchida, Shota Ide, Brian Kenji Iwana, Anna Zhu (2016), “A Further Step to Perfect Accuracy by Training CNN with Larger Data”, 15th International Conference on Frontiers in Handwriting Recognition, 405-410

[4] Li Deng (November 2012), “The MNIST Database of Handwritten Digit Images for Machine Learning Research”, Best of the web series, IEEE signal processing magazine, pp. 141-142.

Thank You

Sudipta Ghosh | Department of Mathematics, IIT Kharagpur