INTERNSHIP: PROJECT REPORT

Name of The Student	**Christina Sharanyaa Selvakumari S**
Project Title	Automate identification and recognition of handwritten text from an image.
Name of the Company	TCS iON
Name of the Industry Mentor	Ashutosh Tiwari
Name of the Institute	Panimalar Engineering College Chennai City Campus

Start Date	End Date	Total Effort (hrs.)	Project Environment	Tools used
16-Dec-2024	16-Mar-2025	210	PyCharm IDE	OpenCV, Numpy, Keras, Scikit-learn, Tensorflow, Matplotlib etc.

Project Synopsis:

The primary goal of this industry project is to develop machine learning algorithms to enable entity and knowledge retrieval from documents containing numerical data. The document can be a scanned document, a photo of a document, or an uploaded file. This project aims to develop a computer vision system to extract numerical values from an image efficiently.

Optical Number Recognition (ONR) is a field of research in pattern recognition, artificial intelligence, and computer vision. ONR is used to extract digital data from a physical entity, such as invoices, bank statements, receipts, business cards, emails, or any suitable documentation. ONR allows the creation of a digital copy of numerical data so that it can be edited, searched, and stored efficiently. The extracted data can further be used in processes like cognitive computing, machine learning/translation, text-to-speech applications, and data mining.

This project focuses on identifying numerical values from an uploaded image using various Python packages and functions.

Assumptions:

- The numerical data should be digits between 0 and 9.
- The text across the input image must be clear with no noisy background.
- The numerical data should be black on a white background.
- • All machine dependencies must be correctly installed.

Project Diagrams:

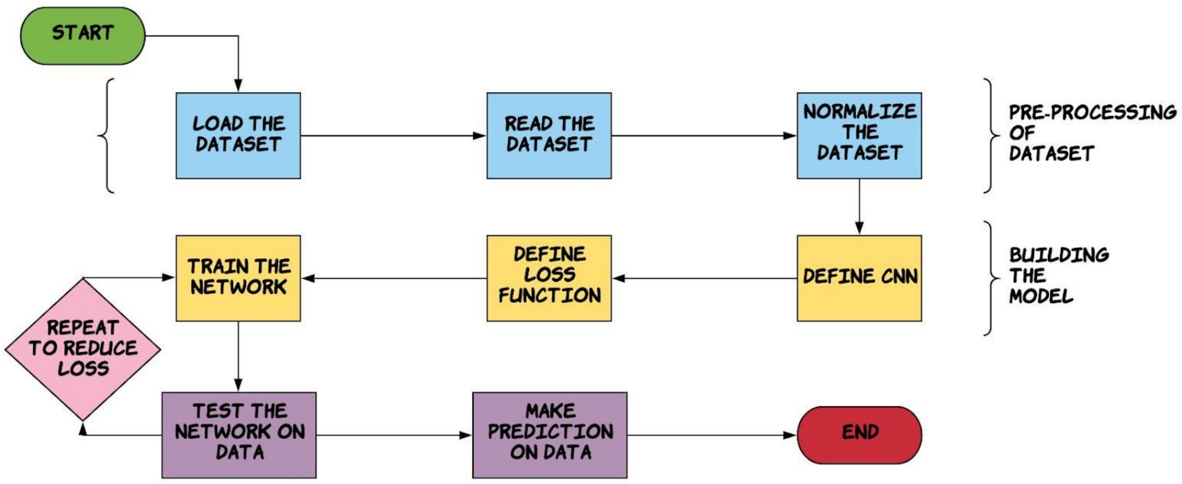


Figure 1. Convolution Neural Network (CNN)

Algorithms:

Optical Number Recognition (ONR) technology identifies numerical data inside images, such as scanned documents and photos. ONR converts images containing written numbers (typed, handwritten, or printed) into a digital format.

Text identification is a two-step process:

1. **Spot the text in the image.**
2. **Identify the characters.**

Main Approach

The primary method used in this project is the Convolutional Neural Network (CNN).

Architecture:

- **CNN Layers:**
  - Convolutional layer with 5x5 filter kernels in the first 2 layers and 3x3 in the last 3 layers.
  - RELU function
  - Pooling Layer
  - Output feature map

Data:

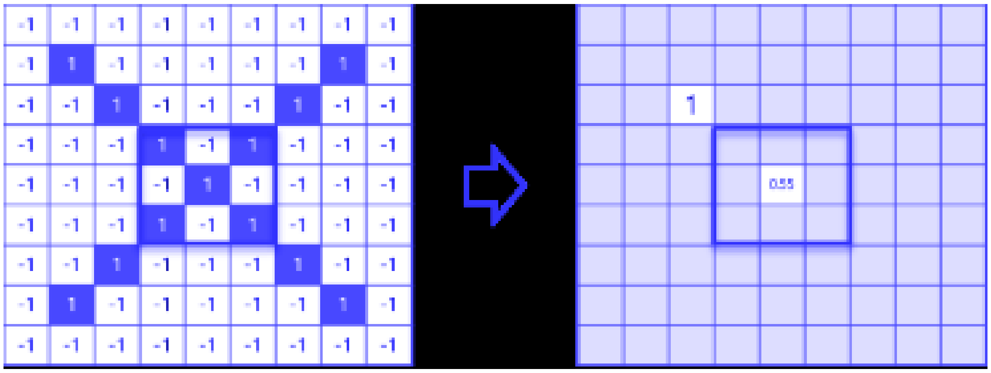
- Input: A gray-scale image of size 32x32
- Output: Character-probability Matrix

b. **Process:**
c. Create CNN layer and return an output.
d. For each layer, create a convolution kernel of size k×k.
e. Feed the convolution result into the RELU operation and to the pooling layer with size px×py and step-size sx×sy.
f. Repeat the above steps for all layers.
g. Train the Neural Network using the mean of the loss values of the batch.
h. Feed the values into an optimizer such as RMSProp.

Working:

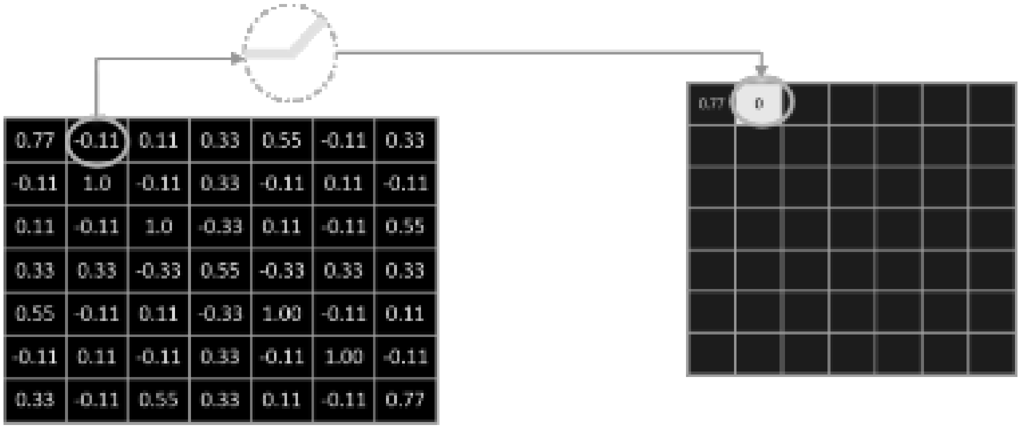
1. **Convolution:**

Convolution is translational invariant, meaning each convolution filter represents a feature of interest (like pixels). The CNN algorithm learns which features form the target reference. The output strength is independent of the feature's position, so the CNN can identify a number even if it's located in different parts of the image.



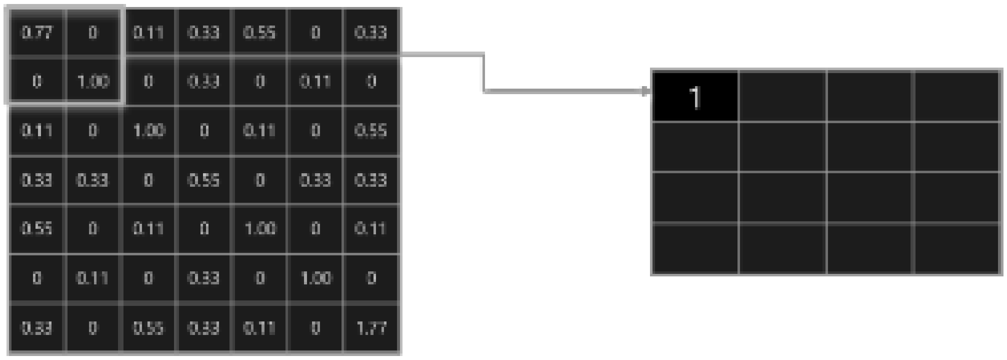
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

2. **Rectified Linear Unit (RELU):**
The RELU function activates a node only if the input surpasses a certain value. If the input is below zero, the output is zero. If the input exceeds the threshold, it forms a linear relationship with the dependent variable. This removes negative values from the convolution output.



0.77	-0.11	0.11	0.33	0.55	-0.11	0.33
-0.11	1.0	-0.11	0.33	-0.11	0.11	-0.11
0.11	-0.11	1.0	-0.33	0.11	-0.11	0.55
0.33	0.33	-0.33	0.55	-0.33	0.33	0.33
0.55	-0.11	0.11	-0.33	1.00	-0.11	0.11
-0.11	0.11	-0.11	0.33	-0.11	1.00	-0.11
0.33	-0.11	0.55	0.33	0.11	-0.11	0.77

3. **Pooling Layer:**
   This layer reduces the size of the image stack. Pooling is done after passing through the activation layer.

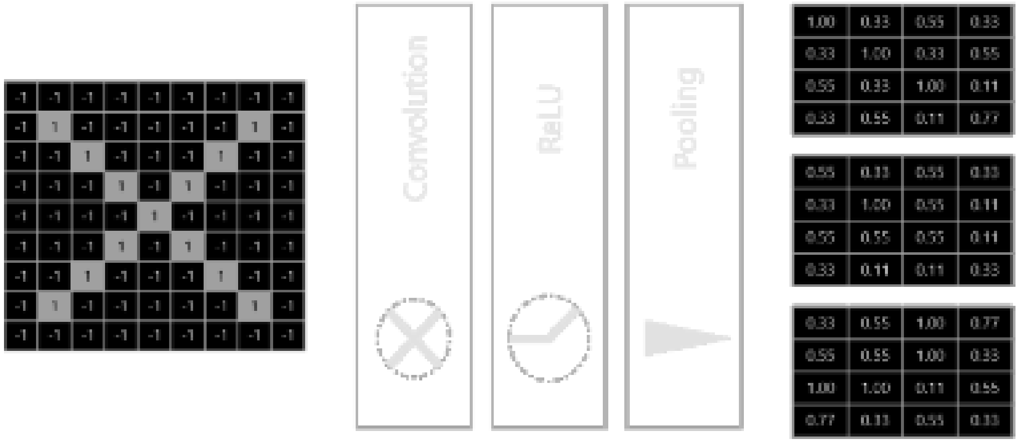


0.77	0	0.11	0.33	0.55	0	0.33
0	1.00	0	0.33	0	0.11	0
0.11	0	1.00	0	0.11	0	0.55
0.33	0.33	0	0.55	0	0.33	0.33
0.55	0	0.11	0	1.00	0	0.11
0	0.11	0	0.33	0	1.00	0
0.33	0	0.55	0.33	0.11	0	1.77

1			

4. **Stacking Layer:**
   The final layers in the network are fully connected — neurons from the preceding layers are linked to all neurons in subsequent layers.

   4x4 matrix from a 7x7 matrix after Pooling:



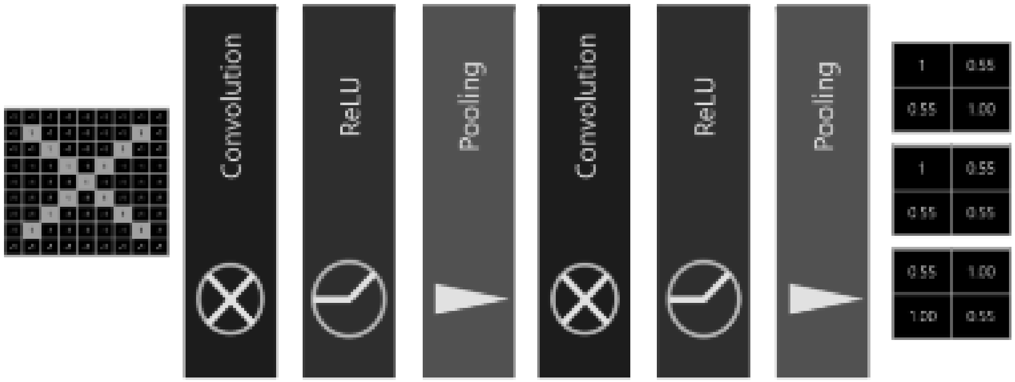
-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

1.00	0.33	0.55	0.33
0.33	1.00	0.33	0.55
0.55	0.33	1.00	0.11
0.33	0.55	0.11	0.77

0.55	0.33	0.55	0.33
0.33	1.00	0.55	0.11
0.55	0.55	0.55	0.11
0.33	0.11	0.11	0.33

0.33	0.55	1.00	0.77
0.55	0.55	1.00	0.33
1.00	1.00	0.11	0.55
0.77	0.33	0.55	0.33

Reduce the image from 4×4 to 2x2 by performing the 3 operations in iteration after the first pass. So after the second pass we arrive at a 2×2 matrix as shown below:



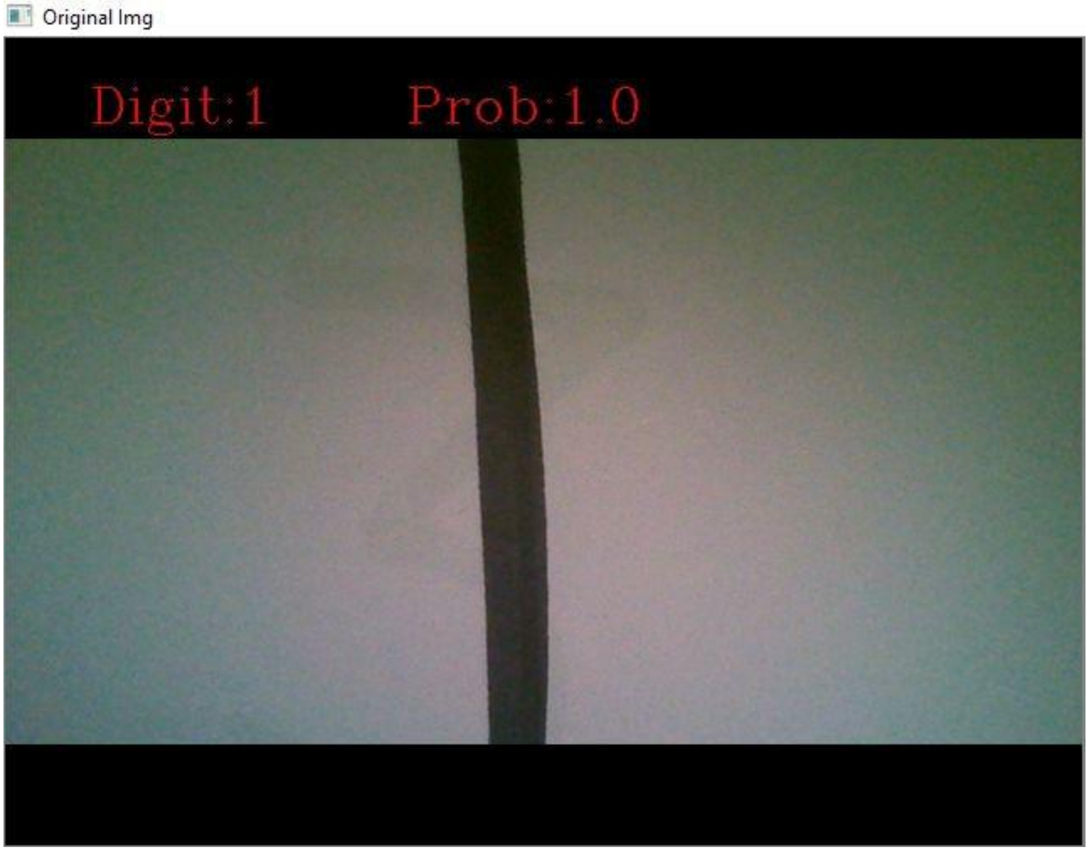
The last layers in the network are fully connected, meaning that neurons of preceding layers are connected to every neuron in subsequent layers.

5. **Estimation:**
Once the network is trained, it can begin to estimate and validate the classifier's performance.

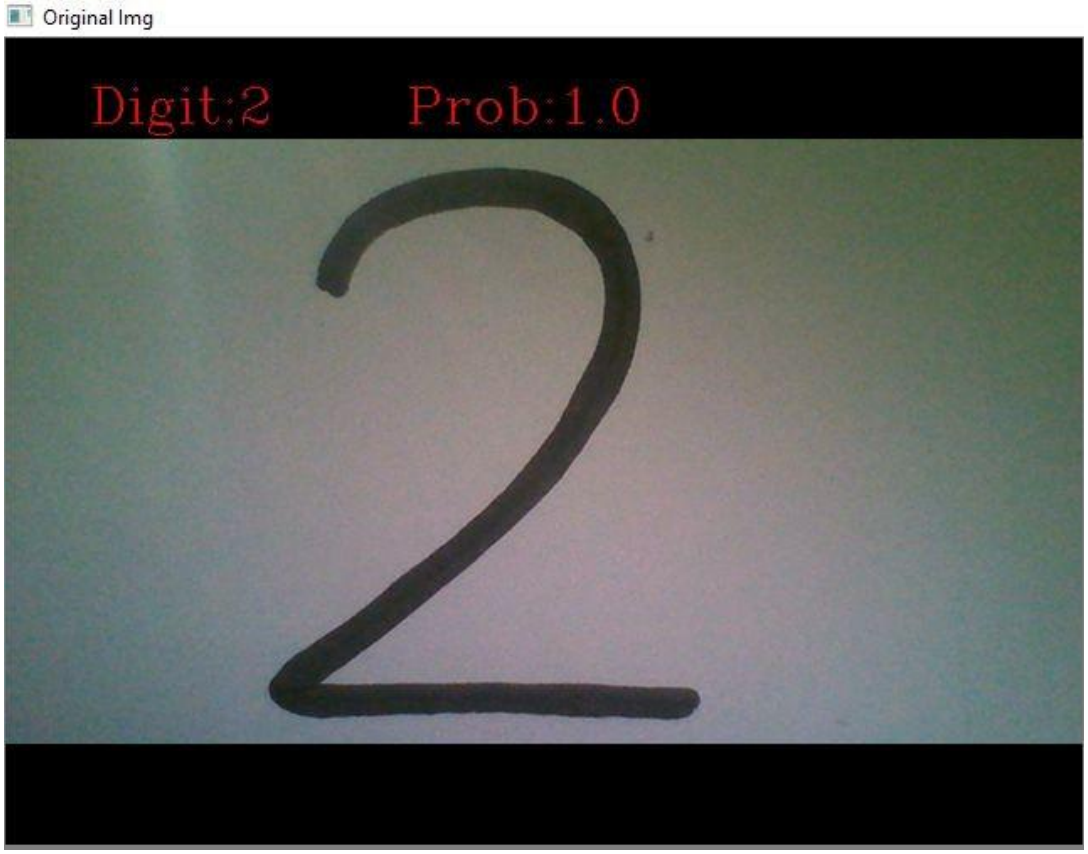
Outcome:

The algorithm can identify numerical data from an image with about **95% efficiency** during evaluation.

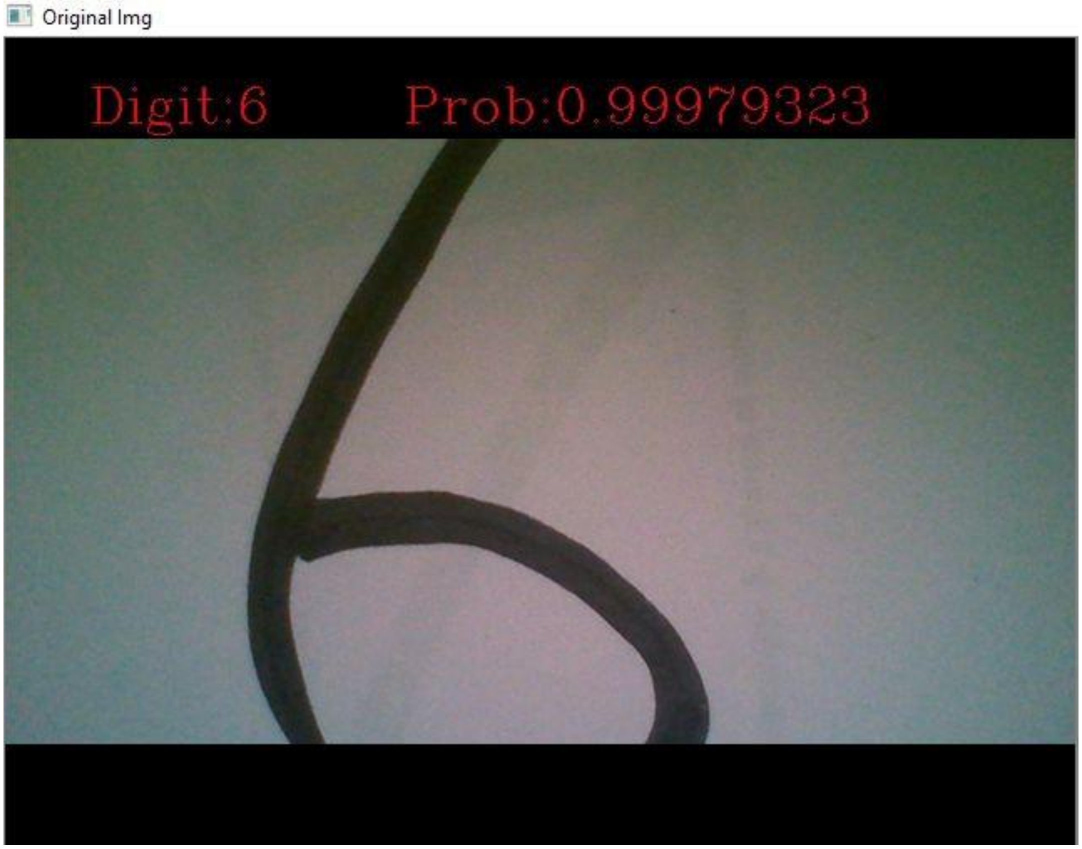
Example Results:
The model successfully identified numerical values with high efficiency, as shown in the examples below:



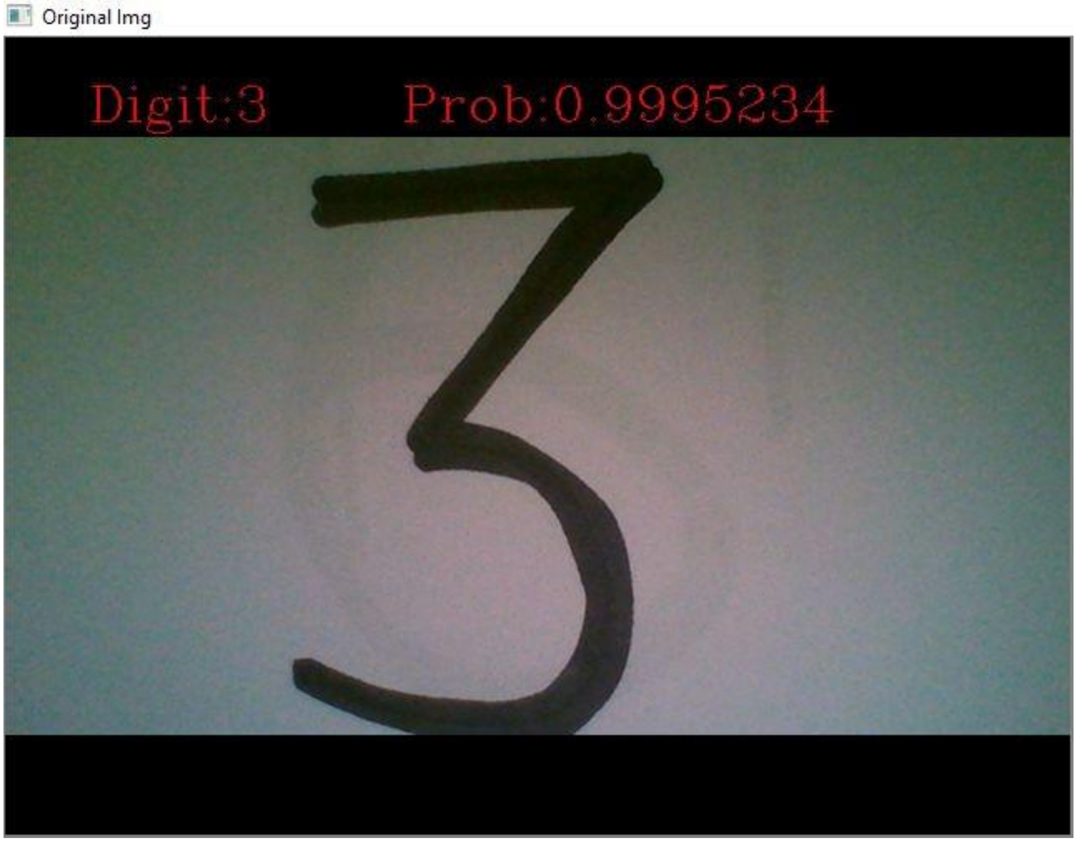
Digit:1 Prob:1.0



Digit:2 Prob:1.0



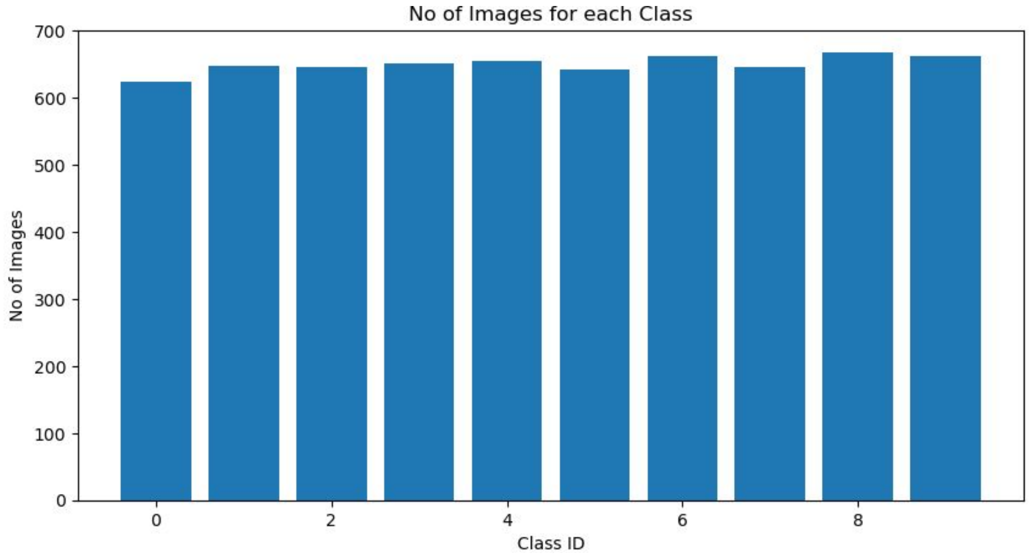
```
Digit:6        Prob:0.99979323
```



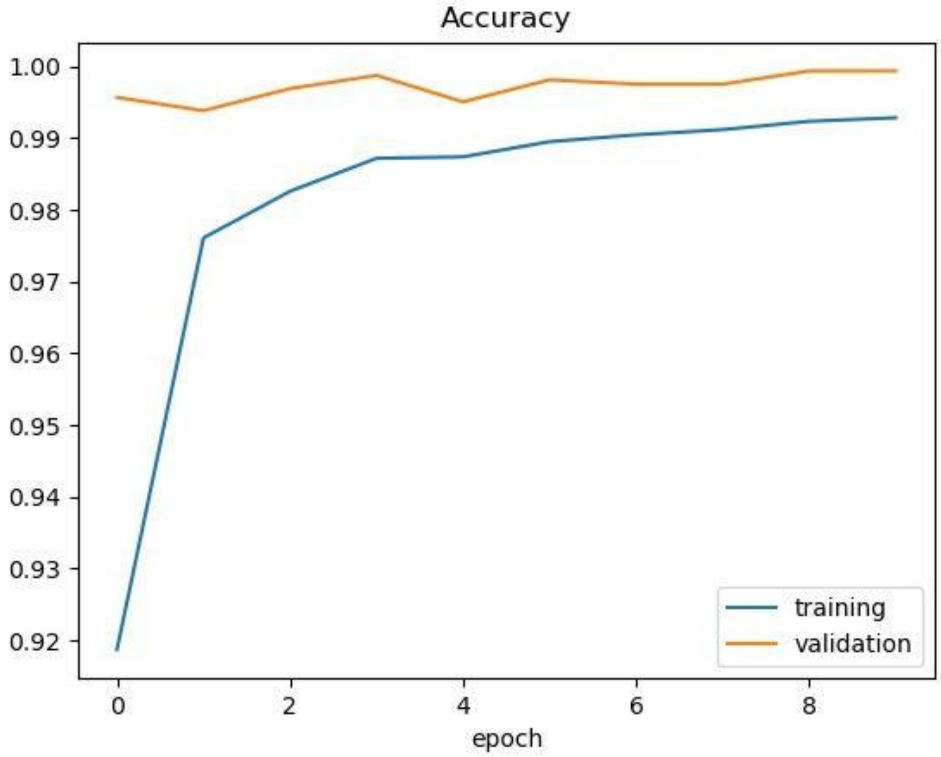
```
Digit:3        Prob:0.9995234
```

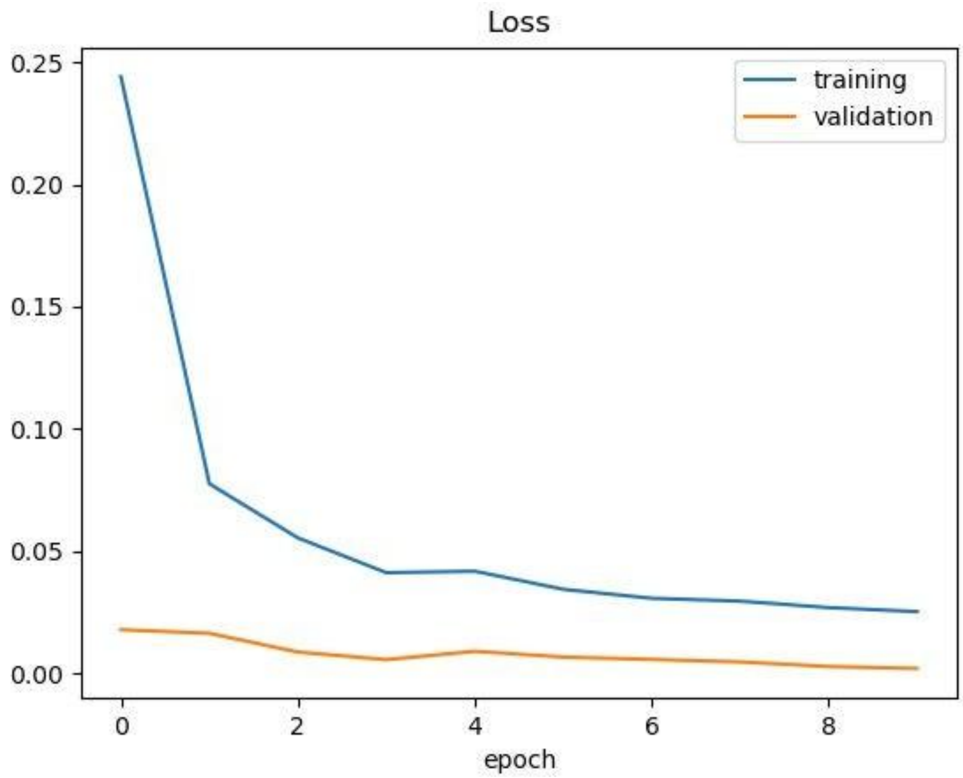
As we can see the model is quite accurate and successfully able to extract the handwritten text.

The distribution of training data used can also be seen through this graph:



Accuracy and Loss function plots are given below:





Exceptions considered:

1. The text in the image must be black on a white background.
2. The background should not contain aggressive markings or patterns interfering with the text.

Enhancement Scope:

1. Efficiency can be improved with predefined models and more powerful machine learning GPUs.
2. Future improvements could include identifying numbers in multiple languages.

References:

https://software.intel.com/content/www/us/en/develop/training/course-artificial-intelligence.html
https://software.intel.com/content/www/us/en/develop/training/course-machine-learning.html
https://keras.io/guides/training_with_built_in_methods/

Link to Code and executable file:
https://github.com/chrissharansel/Handwritten_Num