**MNIST DIGITAL CLASSIFICATION AND HANDWRITTEN DIGIT RECOGNITION**

Team Members: Vinay Kumar Mannava

**TABLE OF CONTENTS**

**1. Objective ----------------------------------------------------------- 3**

**2. Problem Statement ----------------------------------------------- 3**

**3. Solution Overview ------------------------------------------------ 3**

**4. Functional Workflow -------------------------------------------- 3**

**5.** **Architecture Diagram ------------------------------------------- 4**

**6.** **Component Exploration ---------------------------------------- 4**

* Input Interface **-------------------------------------------------- 4**
* Preprocessing & Prompt Creation **---------------------------** **4**
* Embedding Model **--------------------------------------------- 5**
* Chunking Strategy **--------------------------------------------- 5**
* Vector Store **----------------------------------------------------- 5**
* Retriever **-------------------------------------------------------- 5**
* Language Model **----------------------------------------------- 5**
* Support Vector Machine **-------------------------------------- 5**
* Response Generation & Display **----------------------------- 6**

**7.** **Sample Output ---------------------------------------------------- 6**

**7.** **Conclusion --------------------------------------------------------- 7**

**8.** **References --------------------------------------------------------- 7**

**Objective**

The objective of this project is to develop a web-based application capable of recognizing handwritten digits in real-time using a hybrid approach combining Convolutional Neural Network (CNN) and Support Vector Machine (SVM) trained on the MNIST dataset. The application aims to provide an interactive and educational demonstration of machine learning in computer vision.

**Problem Statement**

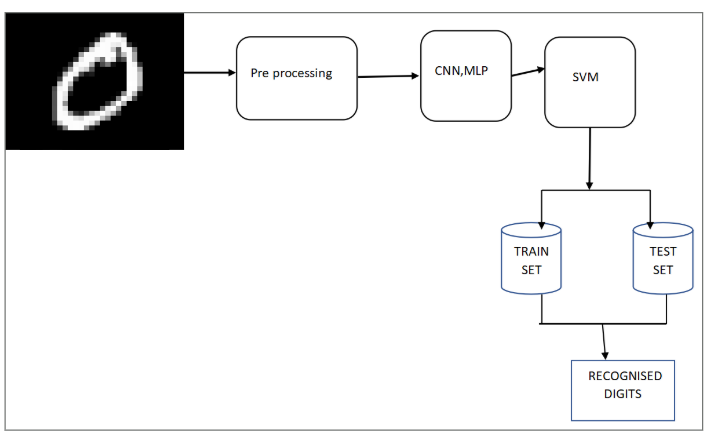
Handwritten digit recognition is a classic problem in computer vision and pattern recognition. Variations in handwriting styles, sizes, and orientations make it challenging for traditional algorithms to accurately classify digits. The goal is to create a robust, user-friendly system that can accurately recognize digits drawn by users on a digital canvas using both deep learning and traditional machine learning approaches.

**SOLUTION OVERVIEW**

This project leverages a hybrid machine learning approach using both CNN and SVM trained on the MNIST dataset. The CNN extracts high-level features from the input images, while the SVM provides additional classification capabilities. The trained models are converted to TensorFlow.js format for browser-based inference. Users can draw digits on a web canvas, and the application predicts the digit using both models, displaying the result and confidence scores.

**Functional Workflow**

* User draws a digit on the web canvas.
* The drawn image is preprocessed and resized to 28x28 pixels.
* The image data is passed to both CNN and SVM models running in the browser.
* CNN extracts features and provides initial classification.
* SVM provides additional classification based on extracted features.
* Results from both models are combined for final prediction.
* The result and a probability bar chart are displayed to the user.

**ARCHITECTURE DIAGRAM** 

**COMPONENT EXPLORATION**

**Input Interface:**

Interactive HTML5 canvas for drawing digits using mouse or touch.

**Prompt Creation:**

Captures the canvas image, resizes to 28x28 pixels, and normalizes pixel values for both CNN and SVM models.

**Embedding Model (CNN):**

A Convolutional Neural Network trained on MNIST, exported to TensorFlow.js for browser inference. The CNN extracts high-level features from the input images and provides initial classification.

**Chunking Strategy (Data Preparation):**

The input image is chunked into a 28x28 array, matching the model's expected input. Features are extracted and prepared for both CNN and SVM processing.

**Vector Store (Model Weights/Storage):**

Model weights are stored in the models/ directory as .json and .bin files, loaded by TensorFlow.js. Both CNN and SVM model parameters are stored and accessed during inference.

**Retriever (Prediction Logic):**

The JavaScript logic retrieves the processed image, feeds it to both CNN and SVM models, and collects the output probabilities from both models for ensemble prediction.

**Language Model (CNN / ML Model):**

The CNN acts as the primary "language model," interpreting the input image and producing a digit classification with high-level feature extraction capabilities.

**Support Vector Machine (SVM):**

The SVM provides additional classification capabilities, working with features extracted by the CNN to improve overall prediction accuracy and robustness.

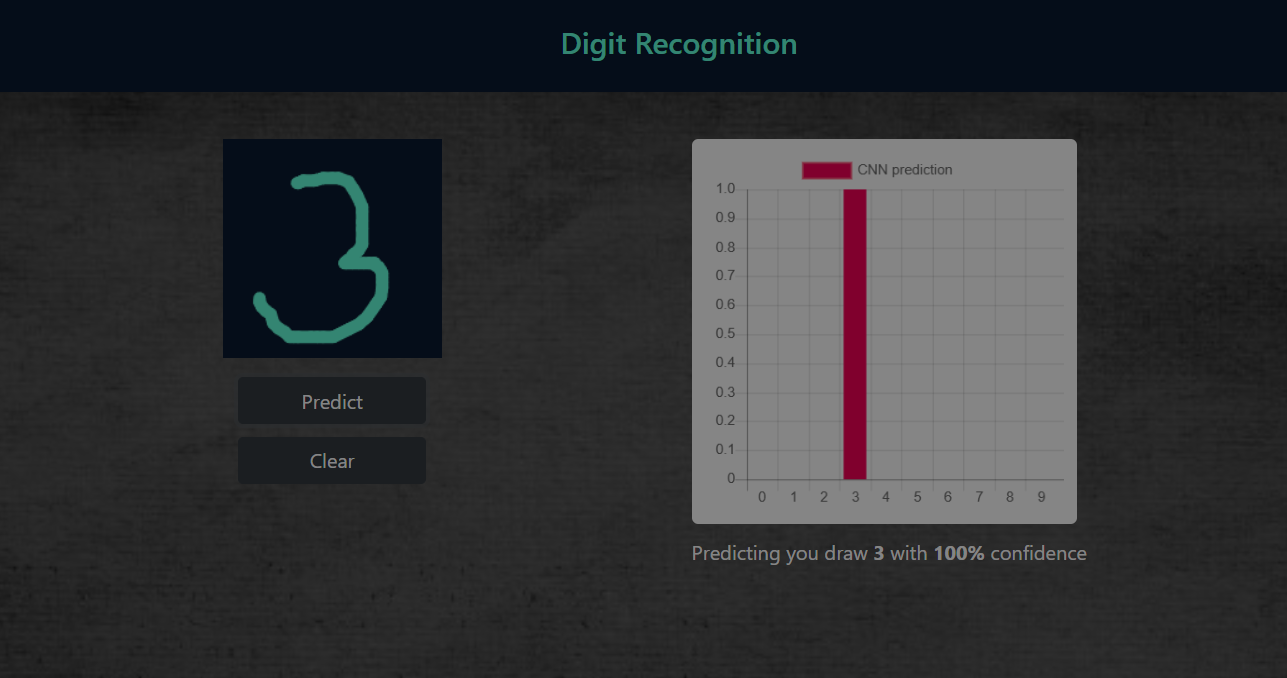
**Response Generation & Display:**

The predicted digit and confidence scores from both CNN and SVM are combined and displayed, with a bar chart visualizing the ensemble probabilities.

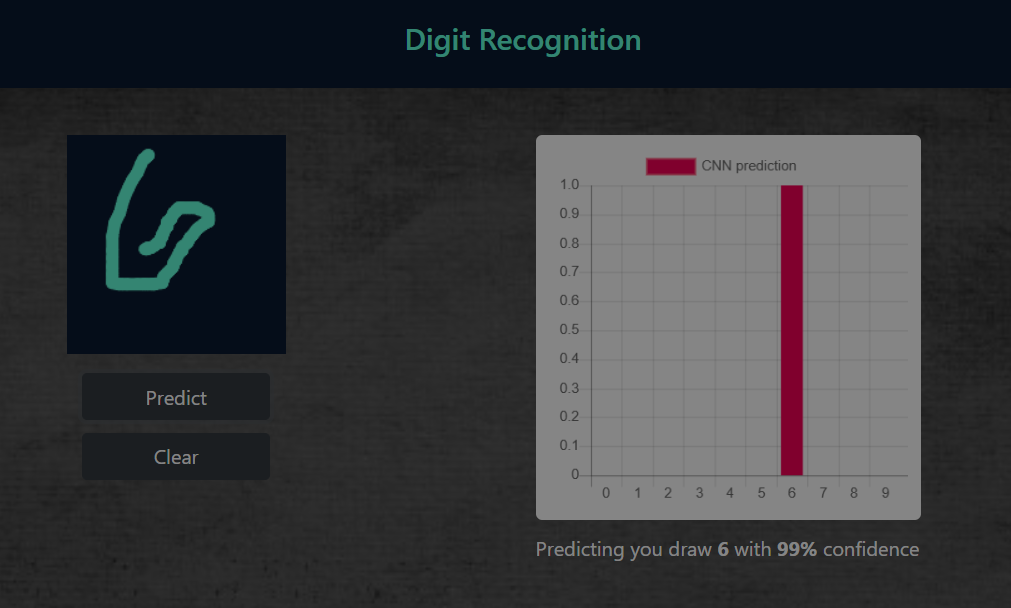
**Sample Output:**

Below are sample outputs from the application:

**Prediction for digit 3:**



**Prediction for digit 6:**



**Conclusion**

This project demonstrates the effectiveness of hybrid machine learning approaches combining CNN and SVM for handwritten digit recognition in a browser environment. The ensemble approach provides improved accuracy and robustness compared to using either model alone. The interactive interface and real-time feedback make it a valuable educational tool and a foundation for more advanced computer vision applications.

**References**

* <https://www.tensorflow.org/js>
* <https://www.chartjs.org/>
* <https://getbootstrap.com/>
* <https://scikit-learn.org/stable/modules/svm.html>
* <https://www.tensorflow.org/datasets/catalog/mnist>