Project Report

Offline Letter & Number Recognition System

Machine Learning Final Project

January 2026

1 Project Overview

The **Offline Letter & Number Recognition System** is a desktop-based application designed to recognize handwritten characters (uppercase letters, lowercase letters, and digits) in an offline environment. The system utilizes a Deep Learning model specifically a Convolutional Neural Network (CNN) to process visual input from a digital canvas and provide immediate textual and auditory feedback.

2 Technical Specifications

- **Programming Language:** Python 3.x
- **Deep Learning Framework:** TensorFlow/Keras
- **GUI Framework:** Tkinter (with themed widgets)
- **Image Processing:** PIL (Pillow)
- **Speech Synthesis:** Pyttsx3 (Offline Text-to-Speech)
- **Dataset:** EMNIST Balanced (47 classes including 0-9, A-Z, and select lowercase letters)

3 System Methodology & Working

The application follows a four-stage pipeline to achieve high accuracy:

3.1 A. Data Acquisition (Canvas Component)

The user interacts with a 300×300 pixel black canvas. Using the mouse or a touchscreen, the user "inks" pixels in white. Simultaneously, a background image (PIL object) mirrors these movements to capture a high-resolution version of the drawing for processing.

3.2 B. Smart Preprocessing

This is a critical stage for accuracy. Instead of simply shrinking the 300×300 canvas to 28×28, the system performs the following:

1. **Bounding Box Detection:** It identifies the exact area containing the ink.
2. **Cropping & Padding:** It crops the character and pads it into a square format to maintain the aspect ratio.

3. **Normalization:** The image is resized to 28×28 pixels and pixel values are scaled to a range of $[0, 1]$.

3.3 C. CNN Inference

The preprocessed image is fed into a trained Convolutional Neural Network. The model architecture includes:

- **Convolutional Layers:** To extract spatial features like edges and curves.
- **Pooling Layers:** To reduce dimensionality and provide translation invariance.
- **Dense Layers:** To classify the extracted features into one of the 47 character classes.

3.4 D. Output Generation

The class with the highest probability is retrieved. The UI updates immediately to display the character and the "Confidence Score." An offline voice engine (Pyttsx3) then announces the result.

4 Key Features

- **100% Offline:** All computations, including machine learning inference and voice synthesis, occur locally on the machine.
- **Professional GUI:** A modern, dark-themed interface built for usability.
- **Smart Centering:** Allows users to draw characters anywhere on the canvas without losing accuracy.
- **Extra Credit Enhancements:**
  - **Integrated Eraser:** Allows the user to correct mistakes on the canvas.
  - **Audio Feedback:** Real-time speech output for accessibility.

5 Setup & Installation

1. Ensure the model file `char_recognition_model.h5` is in the root directory.
2. Install dependencies:

```
pip install tensorflow pillow pyttsx3
```

3. Run the application:

```
python main_app.py
```

6 Conclusion

The system successfully meets all project objectives by providing a robust, fast, and offline solution for handwriting recognition. By utilizing a CNN and intelligent preprocessing, it achieves high reliability across varying handwriting styles.