Iteration 3 Report
EECE 2560: Fundamentals of Engineering Algorithms
Handwritten Digit Recognition

Sean Costello
Department of Electrical and Computer Engineering
Northeastern University
Costello.S@northeastern.edu

November 4, 2025

Important: Each student must submit individually, even though the project is completed as a team. Prepare this report in **Overleaf** and export it as a **PDF** for submission.

GitHub Link: `https://github.com/emescher/DigitRecognition`

Contents

1 Summary of Team Progress and Development Updates

This iteration 3 serves as a status update for our handwritten digit recognition project. Details about the development update and the progress of the team are covered above along with the description of the core features implemented, challenges faced and how they were addressed, and the general timeline/plan. The code can be viewed in our GitHub repository.

During this iteration, we worked on GitHub repository, which was created and shared to serve as the central workspace for code collaboration. This setup ensured that all the Google Colab notebooks and files were efficiently managed and synchronized across team members. Each team member was granted collaborator access and a standardized branching and commit protocol was agreed upon during the group meeting, to maintain proper project organization. For now, all the files and Google Colab notebooks are in the main branch, but another branch has been created and all the data will be moved to the new branch. During this session, the team linked the main Google Colab notebook to the GitHub repository. This meeting established the foundation for the model development and data processing tasks planned for upcoming days. A team leader (Sean Costello) was assigned during this meeting to ensure task distribution and to maintain team communication.

2 Implemented Core Features

Feature 1: MNIST dataset

Purpose: To load, preprocess, and visualise the MNIST dataset for digit recognition.

Implementation:

- The dataset was loaded using TensorFlow and its structure was verified using NumPy and `matplotlib.pyplot` for visualisation.
- `os` was imported (file existence checks), NumPy for array operations, `cv2` for reading and manipulating images, `matplotlib.pyplot` for displaying images, and TensorFlow (`tf`) for loading and running the trained neural network.
- The commented section loads MNIST and normalizes pixel values to [0,1] (required for NN input). It was commented out because the model was already trained and saved.
- MNIST digits are white on black; inversion steps ensure consistency if images are black on white.

Validation: Sample digit images were displayed using `matplotlib.pyplot`, ensuring that the dataset was correctly loaded and formatted.

Feature 2: Model training

Purpose: To develop and train the model for the project.

Implementation:

- A three-layer neural network was implemented and trained using TensorFlow.
- The model was trained for 6 epochs (i.e., six full passes through the training data).
- Accuracy was evaluated on the test dataset via `model.fit()` with validation data.

Validation:

- Training metrics were recorded and plots generated; the notebook was pushed to GitHub.
- Final training accuracy reached 99% and validation accuracy reached 97.99%.
- Decreasing validation loss indicated effective learning and digit discrimination.
- This forms a reliable baseline to compare with KNN and SVM in future work (graphs available in the repository's data branch).

Feature 3: GUI design (still in progress)

Purpose: To design a graphical user interface that allows users to draw digits and test the trained model's predictions.

Implementation:

- The GUI was developed in Spyder and includes a drawing canvas plus buttons such as *clear* and *save*.

Validation:

- Implementation is planned for the next iteration. For testing, drawings were manually uploaded: images were sent from one laptop via Outlook and then uploaded to the MNIST pipeline. This allowed testing both the dataset path and the trained model prior to direct GUI–model integration.

3 Challenges and Resolutions

- **Challenge 1:** Difficulty collaborating on GitHub without overwriting each other's work.
  Resolution: Held a brief meeting and learned how to create separate branches; committed changes frequently. This ensured Google Colab notebooks and files were efficiently managed and synchronized.
- **Challenge 2:** Understanding and integrating the MNIST dataset into the project.
  Resolution: Discussed preprocessing and model testing strategy. Through research and group discussions, we connected the dataset to the trained model and verified functionality.
- **Challenge 3:** Designing the graphical user interface (GUI) layout.
  Resolution: Met to discuss structure and divided tasks. By collaborating and testing different layout ideas, we designed a functional interface to be connected to the trained model later.

4 Leadership Rotation and Team Contributions

Leadership Summary

Week/Span	Leader	Responsibilities	Key Outcomes
Week 1	Sean Costello	Managed team communication and project setup; led kickoff; verified MNIST dataset and model initialization	Integrated MNIST dataset and successfully initialized the model

Individual Contributions

Team Member	Contributions (Technical / Documentation)	Hours
Sean Costello	Served as project manager; handled communication, scheduling, and MNIST dataset setup; Built neural network architecture; implemented baseline model code	5 hrs
Eva Mescher	Designed the GUI layout that allows drawing digits and saving them as PNG files for model inference; participated in system testing and group meetings; organized GitHub repository	5 hrs
Sara Sethi	Trained the model and evaluated accuracy; wrote meeting notes; organized GitHub repository	5 hrs

Statement by the Individual Submitter

I, **Sean Costello**, confirm that the above table accurately reflects my personal contributions during Iteration 3.