Final Report

Abstraction

The goal of our project, led by Team MENJ (Mahmud Hasan, John Royal, Nur Mazumder, Erlin Paredes), was to create a system capable of recognizing text. We aimed to achieve this by using machine learning models to identify handwritten digits.

Introduction

Our project utilized the MNIST dataset, which contains 60,000 training images and 10,000 testing images of handwritten digits. These images are converted into a black and white, 28x28 pixel format, providing a standardized dataset for our models.

Pre-processing

The pre-processing stage involved preparing the MNIST dataset for our models. This included converting the images into a format that our models could interpret and use for training. In the case of the neural network, the image's pixel data was converted from its black and white 0 to 255 values into a 28x28 matrix and then a 784x1 matrix to be compatible with the model.

Modeling

We used several models in our project:

- K-means
- K-nearest-neighbors
- Logistic regression
- Neural Network

Each model was used to train on the MNIST dataset and test its ability to accurately recognize the handwritten digits.

Analysis

Our analysis focused on comparing the performance of each model.

Mahmud Hasan was responsible for the Neural Network model, which was built from scratch using mostly numpy. The neural network model built using PyTorch was created similarly to the one built from scratch. It has around 85% accuracy, but can be improved with addition of more nodes and hidden layers. It does not seem to be overfitting as it has similar accuracy when testing images it had not seen before. An autoencoder network was also built to attempt to recreate images into the MNIST database, which may help train a neural network that can run on less data. Additionally, there is a generator which attempts to use the neural network model in reverse and create new handwritten digits of a similar type and style as the MNIST database, however it is still very inaccurate.

Project Organization

The final scripts, libraries, and key notebooks used in our project include:

- Libraries: Pandas, NumPy, Matplotlib, Scikit-learn
- Models: K-means, K-nearest-neighbors, Logistic regression, Neural Network
- Key Notebooks: neuralnet.py (contains the code for the Neural Network model)

Summary

Our project successfully created a system capable of recognizing text using various machine learning models. While there were challenges along the way, such as the inaccuracy of the generator, we learned valuable lessons about the application of machine learning in text recognition.

Team Contributions

Quantitative contributions to the project are as follows:

- Mahmud Hasan: Developed the Neural Network model, including the autoencoder network and the generator.
- John Royal:
- Nur Mazumder:
- Erlin Paredes:

https://github.com/QuodFinis/Final-Project/tree/main

CSC44800 Final Project Requestment(CCNY)

The following is what you need to turn in for your project. Please note that these are general requirements and in specific cases, not all of these requirements may apply. If you are not sure about what to include, please consult with me before your submission.

Research Group and Github

I will list the group in the group division file.

Each group needs to build a final project github repository, you need to organize the repository well, the organization of the repository is one of the point standards. you need to include a Readme file which describes the task of each member does.

For example, you should have documents folder to put documents; code folder to put codes. Under each folder, build a subfolder for each member, put each member's work under the member's name, also put the summary together.

Your project distribution files should contain the following:

1. Complete source code (**be sure that your source code is fully documented and easy to read**).
2. Binary files (e.g., executables, DLLs, Class files) or other components necessary to run your program.
3. Readme file containing instructions on how to compile, install, and/or run your program.
4. Project report/documentation, as described above.
5. Any test data used for evaluation of your system.
6. If your application is CGI-based or otherwise has a server component, please provide a URL for a demo version of your system.

Implementation

The Implementation projects will be evaluated based on significance, design, correctness, documentation, and appropriate evaluation/testing. You will need to electronically submit a compressed file containing your project distribution files and documentation. Your project documentation should contain the following components:

- The problem you are solving
- the data you are you using
- what you accomplished (highlights)*
- project organization (which are the final scripts, libraries, and key notebooks)
- who did what on the team quantitatively (not just qualitatively)

Final Report

1) Abstraction: what the goal of your project? What achievement you get?
2) Introduction: Introduce the background, model information stuff
3) Pre-processing
4) Modeling
5) Analysis
6) Summary