Multimodal ML Project Group
Lester Bonilla, Nathan Gilman
Project Final Report to Intro to Multimodal Machine Learning, CIS 4930
Spring 2023

1. Introduction

1.1 Motivation

The goal of our final project is to implement the beginnings of a machine learning model that can be used to help teach how to write, speak, and understand English words. Our model will start simply - with written and spoken numerals. We will train the model to detect correctly written and spoken digits. This would be useful in a language learning context to provide feedback to students who are learning to pronounce and write numbers. Multimodal machine learning works the same as human multimodal learning. Learning a language in multiple dimensions - written and spoken - will be beneficial to the learning process.

1.2 Background

MNIST Dataset with deep learning:
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9396870
In this paper, the authors utilize Neural Networks to work on the MNIST dataset. They use the Artificial Neural Network and Convolutional Neural Network algorithms for their models. They went through the same preprocessing steps that we've been learning in class. We have not practiced with neural networks in class, so we took this paper as an opportunity to learn how others are using neural networks, and used it as inspiration for how we would set up our data. This gave us a benchmark for performance that we expected to see from our model. In fact, the authors state that the neural networks would be better than the classification models we use, and our models perform comparably to the neural networks.

Spoken digit with tree classifier:
https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5402575
The authors in this paper use a Tree distribution classifier with Mel Frequency Cepstral Coefficients and Vector Quantization. Like the other dataset, we used this as a reference for the possible accuracy of our dataset. The model developed in this paper performed very well, and it gave us a number to keep in mind while developing our models. This paper links a separate paper in which neural networks were used for the same purpose. The neural networks performed slightly worse than the Tree classifier. We considered using neural networks since they seem to perform very well, but we decided that using the classifiers demonstrated in class would be most efficient.

2. Dataset

Our written digit dataset comes from the MNIST data set. This is a free to use dataset of 70,000 images of written digits in various handwritings. We chose this dataset because it is standardized into 28x28 pixels and has many variations for each digit.

The spoken digit dataset is made up of 3,000 recordings between 6 speakers in English. Each digit is spoken 50 times per speaker. We chose this dataset because it was free to use and provided the exact data we were looking for. Having so many variations of numbers gives us a good amount of data to train our models with.

Neither of our datasets require consent forms for use.

3. Analysis

3.1 Exploratory Data Analysis

While conducting our exploratory data analysis, we were happy to find the datasets chosen were already prepped in terms of data cleaning. The .wav files were of the same length and sampling rates and the image files were size normalized and centered. This allowed us to place more focus on feature extraction and the implementation of our model.

The MNIST data set analysis showed that there is an equal distribution of digits in the training and testing sets. The model will be trained and tested on the same proportion of digits. The distribution of numbers is about equal with slightly more ‘1’ samples and slightly less ‘5’ samples. The difference should be slight enough to not make too large of a difference. The model will likely be slightly better at recognizing ‘1’ and slightly worse at recognizing ‘5’.
We plotted a random digit from both the training and testing dataset. We made sure that the label and image matched, and that we had the appropriate amount of data in each set. We plotted a histogram of the amounts of each digit.

For the spoken digit data set, we randomly sampled the data and plotted the waveforms and frequency data. We ensured that we had correctly loaded all 3,000 files.

3.2 Feature Extraction

For our image modality, we used the pixel data as the feature. Each image is a 28x28 vector of values between 0 and 255. We divided each pixel’s value by 255 to scale the values between 0 and 1.

For our audio modality, we extracted three features: Mel-Frequency Cepstral Coefficients, Zero Crossing Rate, and Mel Spectrogram. We chose these features because they

were highlighted and discussed with the course material. More importantly, however, they are critical to most audio recognition models.

After collecting the single modality features, we combined them together. Because we had much more written data than spoken data, we used random sampling to fill in the missing data. Our combined feature vector had the dimensions: (60,000 x 819).

4. Experiment and Results

4.1 Model Architecture

Our final project is a multi-class problem of identifying which of 9 digits a given row of data is. This means that not all classifiers will work appropriately, especially those that operate in a binary fashion. For example, we could not use logistic regression in this case. The classifiers that we did decide to use for our project were Random Forest, K-Nearest Neighbors, Support Vector, and Naive Bayes. We wanted to examine the differences in how performant these models were for our problem context, allowing us to choose the most suitable classifier.

4.2 Experimental Setup

In our experiment, we decided to divide the data into an 80-20 split for training and testing. This is a generally optimal split for the model to train on, while giving good feedback for testing. Many popular resources online recommended using this split.

Working between two machines when developing this project led to slightly different hardware builds. One laptop was a Dell Inspiron with a 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz 1.69 GHz with 16.0 GB of installed RAM. There was no installed GPU. This laptop was additionally working on Windows 11. The other laptop working on this project was an HP Pavilion with a 11th Gen Intel(R) Core(™) i7-1195G7 @ 2.9GHz with 16.0 GB of installed RAM . This laptop was working on Windows 11. Similar to the first laptop, there was no installed GPU beyond the integrated graphics unit.

The version of python that we were using was Python 3.9.16. The version of conda that was being used was Conda 23.1.0. We utilized many packages for python, such as librosa and idx2numpy, that we installed through the conda terminal and pip. We developed our project using Jupyter Notebooks as a way to neatly organize and present our code.

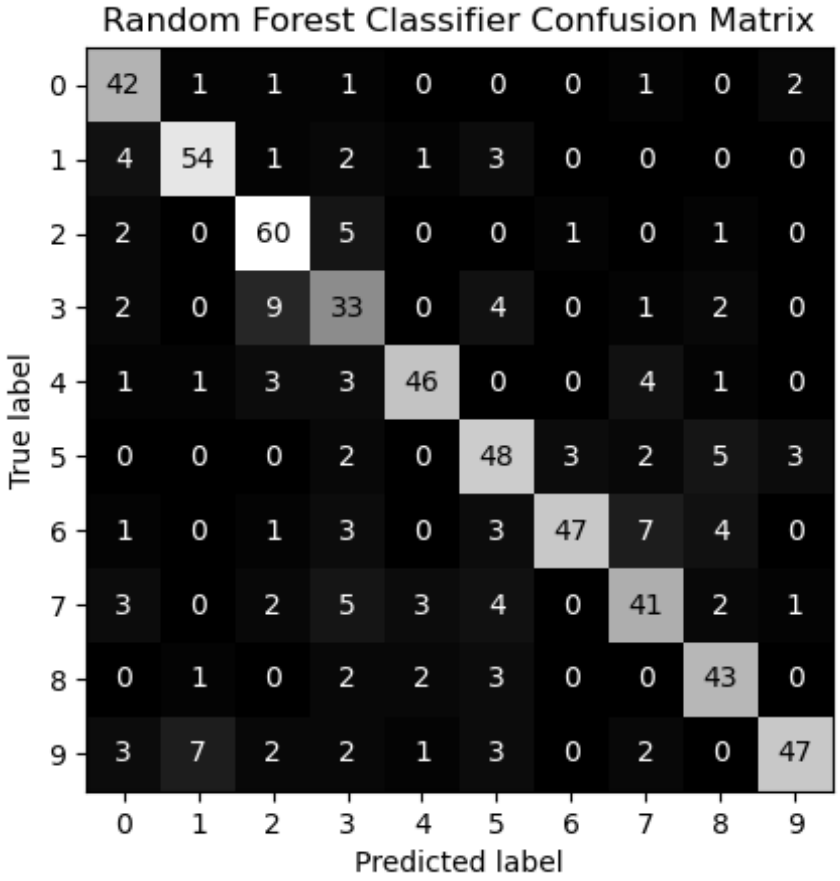
4.3 Results

Prior to creating a fusion layer for our two modalities, we tested how the created models performed on a singular modality. We ran multiple classifiers on each modality to determine which yielded the optimal output. Out of the support vector machine, random forest, naive bayes

and k-nearest neighbors classifiers, the random forest classifier performed the best on the audio modality. The performance metrics for the random forest classifier are below.

	precision	recall	f1-score	support
0	0.72	0.88	0.79	48
1	0.84	0.83	0.84	65
2	0.76	0.87	0.81	69
3	0.57	0.65	0.61	51
4	0.87	0.78	0.82	59
5	0.71	0.76	0.73	63
6	0.92	0.71	0.80	66
7	0.71	0.67	0.69	61
8	0.74	0.84	0.79	51
9	0.89	0.70	0.78	67
accuracy			0.77	600
macro avg	0.77	0.77	0.77	600
weighted avg	0.78	0.77	0.77	600

Furthermore, we generated a confusion matrix for each of the models to see the distribution of the model's predictions to the actual digits expected. The confusion matrix for the random forest classifier is below.



Out of the support vector machine, random forest, naive bayes and k-nearest neighbors classifiers, the random forest classifier and k-nearest neighbors classifier performed almost identically as the best classifiers for the visual modality. They both had an accuracy of 97%. The performance metrics for the two classifiers are below.

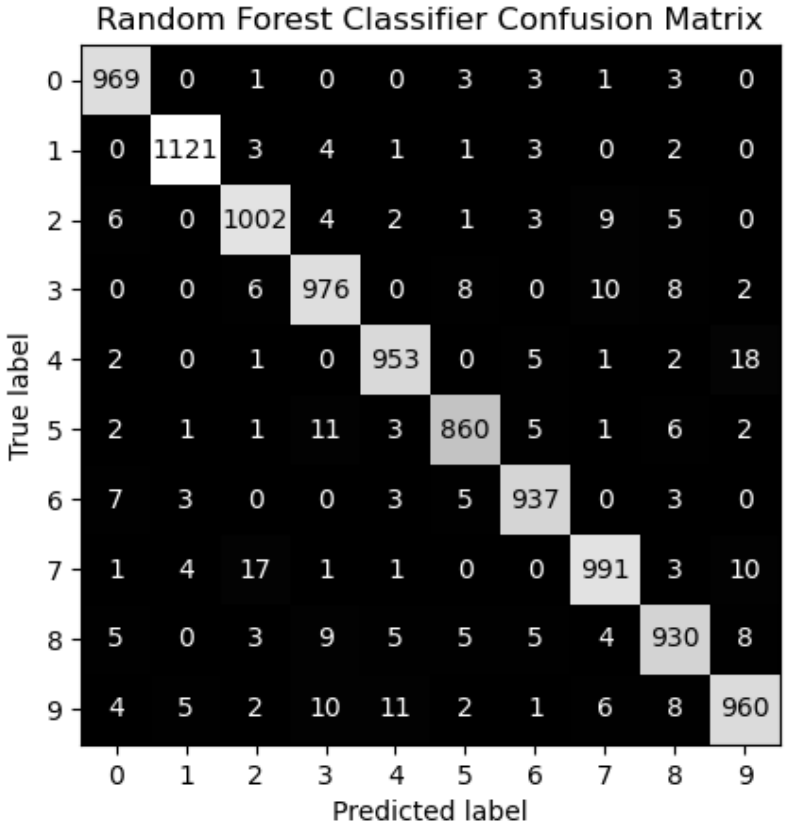
Random Forest

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.97	0.97	0.97	1032
3	0.96	0.97	0.96	1010
4	0.97	0.97	0.97	982
5	0.97	0.96	0.97	892
6	0.97	0.98	0.98	958
7	0.97	0.96	0.97	1028
8	0.96	0.95	0.96	974
9	0.96	0.95	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

K-Nearest Neighbors

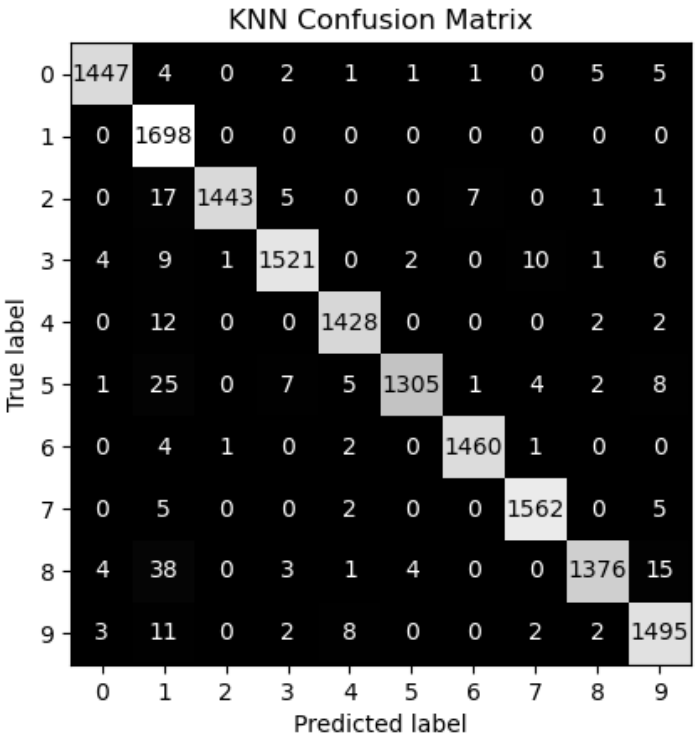
	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.96	1.00	0.98	1135
2	0.98	0.97	0.97	1032
3	0.96	0.97	0.96	1010
4	0.98	0.97	0.97	982
5	0.97	0.96	0.96	892
6	0.98	0.99	0.98	958
7	0.96	0.96	0.96	1028
8	0.99	0.94	0.96	974
9	0.96	0.96	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Similar to our testing for the audio modality, we generated another confusion matrix for each of the models to see the distribution of the model's predictions to the actual digits expected. The confusion matrix for the random forest classifier is below.



After creating the fusion matrix for our datasets, we ran several more tests. And concluded that the K-Nearest Neighbors yielded the most accurate approach with a multimodal dataset. The performance metrics and confusion matrix for the K-Nearest Neighbors classifier is below.

	precision	recall	f1-score	support
0	0.99	0.99	0.99	1466
1	0.93	1.00	0.96	1698
2	1.00	0.98	0.99	1474
3	0.99	0.98	0.98	1554
4	0.99	0.99	0.99	1444
5	0.99	0.96	0.98	1358
6	0.99	0.99	0.99	1468
7	0.99	0.99	0.99	1574
8	0.99	0.95	0.97	1441
9	0.97	0.98	0.98	1523
accuracy			0.98	15000
macro avg	0.98	0.98	0.98	15000
weighted avg	0.98	0.98	0.98	15000



Similar to the K-Nearest Neighbors, all of the other tested classifiers significantly improved in accuracy when using the multimodal dataset. The random forest classifier had an accuracy of 0.97. The support vector classifier had an accuracy of 0.91.

One point of note was that, for each test, the Naive Bayes classifier remained much lower than the other classifiers. For example, the Naive Bayes classifier had an accuracy of 0.65 when trained via our multimodal implementation.

4.4 Error Analysis

Our written model had wrong classifications between digits that look similar. For example, it confused (4,9) and (2,7). Because these digits are written similarly, the variation in handwriting could lead to some digits looking just like another. From the confusion matrix for K-Nearest Neighbors in our multimodal tests, we can see that the digit one is the most common false prediction.

Some of the wrong results on the spoken digit model can be explained by the fact that some digits sound like each other. For example, the model had a higher rate of confusing ‘two’ and ‘three’ and ‘six’ and ‘seven.’ The features extracted from the audio of these words will be similar, causing the model to be confused.

5. Discussion

5.1 Model Performance

Our multimodal model performed better than the single mode models. Combining the features seems to yield a model that was satisfactorily accurate. The model for spoken digit recognition performed insufficiently, likely due to the limited data set. However, with the added aid of the written digit features, the accuracy improved. This performs well for our intended use of the model. The model will always take in both written and spoken digits, so it will always have the aid of the written digit.

5.2 Limitations and Future Work

Because our model was trained on a specific formatting for written digits, it will be limited in identifying digits that fall outside of the 28x28 image range. For practical application, this does not limit its use because we can place restrictions on the input to make sure it is always of the correct format.

Although our written dataset includes a wide range of handwriting styles, we can further increase the accuracy of our written digit recognition by increasing the data set with cursive lettering or other stylizations. Again, for our purposes this does not limit the model's use. The model would take in as input attempts at copying model examples. Because the input would be a replication of the type of fonts the model was trained on, its performance will not be impacted.

Another limitation of the model is the relative lack of unique voices for our spoken set. The set consists of 6 speakers, which is not enough to cover the wide range of English dialects and accents. This can be improved by increasing our data set with a wider range of voices with different dialects and accents.

6. Conclusion

The objective of this study was to develop a model that would be the starting point for a model to be used in a language learning environment. The environment would consist of prompts for the user to write and speak a displayed digit. The model would then analyze the users input, and provide feedback as to how accurately they spoke and wrote the digit. An advanced implementation of this model would then tell the user how they could improve their inputs. Our developed models show a lot of promise for our intended application. Combining the two modalities increased the accuracy of detecting both written and spoken digits.

7. Collaboration Report

Since our team consists of two members and we are handling two modalities, we split up the work by modality. We each did the data analysis and feature extraction for our modality, and then merged them together for model training. We managed the source code using GitHub. We tracked tasks and bugs in real time, we coded the whole project side by side and assisted each other in person. Design and merging decisions were made as they arose. Our team met multiple times to code together and work out the details of our project. This report was made in Google Docs, and we similarly wrote it collaboratively in person. Our teamwork unfolded as we planned. We communicated our availability and each worked with a sense of ownership in the project.

8. Appendix

1. Source Code: https://github.com/npgilman/CIS4930_FinalProject
2. MNIST Data: http://yann.lecun.com/exdb/mnist/
3. SpokenDigit Dataset: https://github.com/Jakobovski/free-spoken-digit-dataset