Signature Verification Project - Final Report

2238-INSY-5378-001-Data Science: A Programming Approach

Professor Mahyar Sharif Vaghefi

Team 2:

Elise Mbungkah

Ashok Rai

Brian Tran

Jay Vikas Warke

December 5, 2023

Table of Contents

Introduction

We were tasked with developing a machine learning model that has the ability to accurately verify a person's signature. Throughout this project, in order to ensure that the model can accurately identify between genuine and faked signatures, this ambitious project will use deep learning techniques, advanced algorithms, and large data sets for model training. Through an exploration of the nuances of signature dynamics, stroke patterns, and personal writing styles, a dependable and safe identity verification solution is the goal of the proposed model. The ultimate objective is to further the development of authentication systems and provide a reliable and effective tool for a variety of applications, such as access control, financial transactions, and legal paperwork, all the while keeping an eye out for potential fraudulent activity. An important dataset containing 80 signature photos from every participant was given to us. This collection serves as the basis for our testing and training your machine learning models, enabling us to create reliable algorithms that can tell the difference between genuine and fake signatures. We were set to build a model that is highly effective at identifying a wide range of writing styles from each participant, so that it can be effectively authenticated in a variety of settings and circumstances.

Data Preprocessing

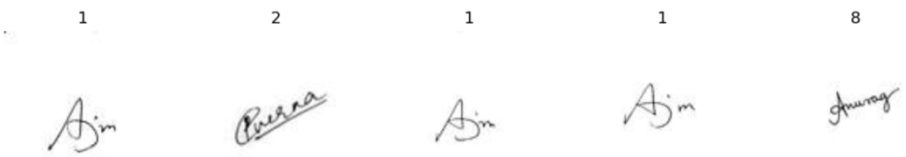
The preprocessing stage is crucial for ensuring the effectiveness of the machine learning model. Key steps in this stage include:

- Format Consistency: Conversion of all signature images to JPG format to standardize the data input.
- Resizing: Standardizing each image to a uniform size of 100x100 pixels, ensuring consistent pattern recognition.
- Cropping: Removal of unnecessary borders around signatures to focus on the essential elements.
- Conversion to Grayscale: Transforming images to grayscale to reduce data complexity and enhance focus on signature patterns.
- The total number of images in the original data was 640, after augmenting the data the final number of images was 3200 which were used in the algorithms discussed below.

To ensure that the cornerstone of a robust machine learning model for signature verification is data preparation. To guarantee correctness and effectiveness in the ensuing analysis, raw data must be carefully and thoughtfully refined. In this paper, we examine the many stages of data preprocessing for signature photos, each of which enhances the model's overall accuracy. First, we pay attention to the photos' format consistency. We establish a common base by requiring all photos to be in JPG format, which makes the subsequent preprocessing processes easier. By creating a common language, this phase makes it easier for the dataset and the algorithm to communicate with each other. Proceeding on, we resize every image to a standard size of 100x100 pixels to address the visual uniformity of the dataset. This makes sure that every

signature takes up the same amount of space, which maximizes the model's capacity to identify and analyze patterns independent of different dimensions. We further refine our method by carefully cropping the photos to remove any unnecessary black frames surrounding the signature. By removing pointless distractions and concentrating the model's attention on the most important aspects, this phase is critical to isolating the signature as the primary factor of interest.

We proceed with the process of streamlining by converting the images to grayscale. This removes potential color variations that could detract from the main analysis of the signature while also simplifying the data. By reducing the amount of information, the grayscale transformation seeks to improve the model's precision in signature recognition. At last, we arrive at an optional phase that adds another level of complexity. This optional step enables subtle customization to satisfy needs and preferences in the analysis of signatures. To sum things up, the process of preparing data for signature verification involves careful iterations and refinement. Every stage helps the machine learning model become more accurate and dependable so that it can distinguish between real and fake signatures. Following an organized preprocessing procedure, we establish the foundation for a model that performs exceptionally well at identifying the fine details of individual signatures, thereby augmenting the overall effectiveness of identity verification systems.



Data Augmentation

To enrich the dataset and improve model robustness, the following augmentation techniques were applied:

- Generation of New Images: For each original signature image, five new images were generated.
- Augmentation Techniques: These include rotation, height and width shift, shear, and zoom, adding variety and complexity to the dataset.
- After data augmentation, the augmented images were combined with the original images

Image Processing

Advanced image processing techniques were employed to highlight signature features:

- Edge Detection: To emphasize the outline and structure of signatures.
- Thresholding: To further isolate and enhance signature details for the model.

Model Training

Four different algorithms were used in the training phase:

1. K-Nearest Neighbors (KNN)
2. Random Forest
3. Multi-Layer Perceptron (MLP)
4. Convolutional Neural Network (CNN)

Prioritizing pixel-based features, using PCA for dimensionality reduction, and standardizing feature values are necessary when building an effective signature detection model. This brief manual emphasizes a simplified method that guarantees a fine-grained comprehension of signature patterns, increases effectiveness in identifying authentic signatures, and preserves uniformity via standardized values. This tactic maximizes the model's capacity to precisely identify complex signature patterns. Important steps for optimum performance are involved in developing a trustworthy signature verification model. The dataset is methodically divided into testing (20%) and training (80%) sets, facilitating efficient model learning and impartial assessment.

Model Evaluation

As the signature verification project comes to an end, the emphasis moves to selecting the best model and getting it ready for implementation. The main procedures are examined in this essay, which involve evaluating each model, choosing the top-performing model, training it on the complete dataset, and preserving the model using the Python pickle package. This short

procedure guarantees that the selected model is both the most efficient and prepared for a smooth transition into practical applications.

The parameters for the 4 different algorithms we have used are as follows:

1. Random Forest

Random Forest is an ensemble learning method renowned for its versatility and robustness in machine learning. The algorithm operates by constructing numerous decision trees during training, each using a subset of the training data and a random selection of features for each split. Through the process of bootstrap aggregating (bagging), where multiple subsets of the training data are created by sampling with replacement, Random Forest mitigates overfitting and enhances generalization. The final prediction in classification tasks is determined by a majority vote of the individual trees, while in regression tasks, it is the average prediction. The ensemble nature of Random Forest allows it to handle complex datasets, providing a reliable model that is less prone to noise and outliers. The algorithm's strength lies in its ability to deliver accurate predictions, assess feature importance, and maintain robustness across various types of data.

Random Forest is a versatile ensemble learning algorithm used for both classification and regression tasks. Here are some general formulas associated with the Random Forest classifier:

Gini Impurity (for Classification)

In decision tree nodes, Gini impurity is a measure of how often a randomly chosen element would be incorrectly classified.

$Gini(t)=1-\sum_{i=1}^{c} p(i|t)^2$

RandomForest Training Data Classification Report:

	precision	recall	f1-score	support
1	0.93	0.96	0.94	384
2	0.97	0.97	0.97	384
3	0.90	0.80	0.84	384
4	0.86	0.73	0.79	384
5	0.94	0.98	0.96	384
6	0.98	0.91	0.94	384
7	0.92	0.90	0.91	384
8	0.75	0.96	0.84	384
9	1.00	0.99	0.99	384
accuracy			0.91	3456
macro avg	0.91	0.91	0.91	3456
weighted avg	0.91	0.91	0.91	3456

The accuracy for training data is 91%.

RandomForest Test Data Classification Report:

	precision	recall	f1-score	support
1	0.82	0.80	0.81	96
2	0.88	0.75	0.81	48
3	0.61	0.46	0.52	96
4	0.58	0.51	0.54	96
5	0.84	0.86	0.85	96
6	0.74	0.73	0.73	96
7	0.49	0.54	0.51	48
8	0.67	0.89	0.76	96
9	0.90	0.97	0.93	96
accuracy			0.73	768
macro avg	0.73	0.72	0.72	768
weighted avg	0.73	0.73	0.73	768

The accuracy for testing data is 73%.

2. K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a simple and intuitive machine learning algorithm used for both classification and regression tasks. At its core, KNN relies on the principle that similar data points tend to have similar outcomes. In the context of classification, when a new data point needs to be predicted, KNN looks at its k nearest neighbors in the feature space, where similarity is often measured using Euclidean distance. The predicted class is determined by the majority class among these neighbors. In regression, the algorithm computes the average or weighted average of the target values of the k nearest neighbors to make predictions.

One of the strengths of KNN is its simplicity and lack of assumptions about the underlying data distribution. However, its performance can be sensitive to the choice of the distance metric and the value of k. Smaller values of k can make the algorithm more sensitive to noise, while larger values can smooth out the decision boundaries. Additionally, KNN does not naturally handle high-dimensional data well, and the computational cost of predicting new instances increases with the size of the dataset.

KNN Training Data Classification Report:

	precision	recall	f1-score	support
1	0.92	0.93	0.93	384
2	0.90	0.88	0.89	384
3	0.69	0.64	0.66	384
4	0.64	0.74	0.69	384
5	0.90	0.89	0.89	384
6	0.94	0.83	0.88	384
7	0.77	0.70	0.73	384
8	0.82	0.93	0.87	384
9	0.94	0.97	0.96	384
accuracy			0.83	3456
macro avg	0.84	0.83	0.83	3456
weighted avg	0.84	0.83	0.83	3456

The accuracy for training data is 83%.

KNN Test Data Classification Report:

	precision	recall	f1-score	support
1	0.89	0.90	0.89	96
2	0.70	0.62	0.66	48
3	0.62	0.47	0.53	96
4	0.52	0.66	0.58	96
5	0.94	0.88	0.91	96
6	0.92	0.79	0.85	96
7	0.49	0.44	0.46	48
8	0.77	0.93	0.84	96
9	0.93	1.00	0.96	96
accuracy			0.77	768
macro avg	0.75	0.74	0.74	768
weighted avg	0.77	0.77	0.77	768

The accuracy for testing data is 77%.

3. Multi-Layer Perceptron (MLP)

The Multi-Layer Perceptron (MLP) is a type of artificial neural network designed for supervised learning tasks, capable of modeling complex relationships and patterns within data. Comprising multiple layers of interconnected nodes, or neurons, MLPs consist of an input layer, one or more hidden layers, and an output layer. Each connection between nodes is assigned a weight, and the model learns these weights during training. Non-linear activation functions, such as the popular Rectified Linear Unit (ReLU) or Sigmoid, introduce non-linearity to the model, enabling it to capture intricate relationships in the data. MLPs are trained using backpropagation, an optimization algorithm that adjusts the weights based on the error between the predicted and actual outcomes. While the architecture and training process make MLPs highly expressive, they

also require careful tuning of hyperparameters and may be susceptible to overfitting, especially on smaller datasets. Despite these considerations, MLPs have demonstrated remarkable success in a wide range of applications, including image recognition, natural language processing, and regression tasks.

NeuralNetwork Training Data Classification Report:

	precision	recall	f1-score	support
1	1.00	1.00	1.00	384
2	1.00	1.00	1.00	384
3	1.00	1.00	1.00	384
4	0.97	0.99	0.98	384
5	1.00	1.00	1.00	384
6	1.00	1.00	1.00	384
7	0.99	0.97	0.98	384
8	1.00	1.00	1.00	384
9	1.00	1.00	1.00	384
accuracy			1.00	3456
macro avg	1.00	1.00	1.00	3456
weighted avg	1.00	1.00	1.00	3456

The accuracy for training is 100%.

NeuralNetwork Test Data Classification Report:

	precision	recall	f1-score	support
1	0.91	0.85	0.88	96
2	0.90	0.79	0.84	48
3	0.84	0.86	0.85	96
4	0.73	0.82	0.77	96
5	0.93	0.90	0.91	96
6	0.87	0.86	0.87	96
7	0.79	0.71	0.75	48
8	0.92	0.94	0.93	96
9	0.94	0.99	0.96	96
accuracy			0.87	768
macro avg	0.87	0.86	0.86	768
weighted avg	0.87	0.87	0.87	768

The accuracy for testing is 87%.

4. Convolutional Neural Network (CNN)

The Convolutional Neural Network (CNN) is a powerful and widely used deep learning architecture designed primarily for processing and analyzing visual data, such as images. It has revolutionized computer vision tasks by automatically learning hierarchical representations of features directly from the raw pixel values. CNNs consist of multiple layers, including convolutional layers, pooling layers, and fully connected layers. Convolutional layers employ filters or kernels to scan input images, capturing spatial hierarchies and detecting patterns like edges or textures. Pooling layers reduce the spatial dimensions of the feature maps, focusing on the most informative elements. These layers enable the network to learn hierarchical representations and spatial hierarchies, making CNNs robust to variations in position, scale, and orientation of objects in images. CNNs have demonstrated exceptional performance in tasks such as image classification, object detection, and image segmentation, making them a cornerstone in the field of computer vision.

The CNN model gives the best result out of the 4 models we have used for the project. We have trained and tested four different models in CNN.

Model 1.

Classification Tree on Training: CNN Model 1

	precision	recall	f1-score	support
0	1.00	1.00	1.00	384
1	0.99	1.00	1.00	192
2	1.00	1.00	1.00	384
3	1.00	1.00	1.00	384
4	0.99	1.00	0.99	384
5	0.99	0.99	0.99	384
6	1.00	1.00	1.00	192
7	1.00	1.00	1.00	384
8	1.00	0.98	0.99	384
accuracy			1.00	3072
macro avg	1.00	1.00	1.00	3072
weighted avg	1.00	1.00	1.00	3072

Classification Report for CNN Model 1

	precision	recall	f1-score	support
0	0.94	0.99	0.96	96
1	1.00	0.98	0.99	48
2	0.98	0.98	0.98	96
3	0.96	0.99	0.97	96
4	0.99	1.00	0.99	96
5	0.95	0.94	0.94	96
6	0.98	0.90	0.93	48
7	0.96	0.98	0.97	96
8	0.99	0.94	0.96	96
accuracy			0.97	768
macro avg	0.97	0.97	0.97	768
weighted avg	0.97	0.97	0.97	768

Model 2.

Classification Tree on Training: CNN Model 2

	precision	recall	f1-score	support
0	1.00	0.99	1.00	384
1	0.90	1.00	0.95	192
2	0.99	1.00	1.00	384
3	0.99	1.00	0.99	384
4	0.99	0.99	0.99	384
5	1.00	0.98	0.99	384
6	1.00	1.00	1.00	192
7	1.00	1.00	1.00	384
8	1.00	0.95	0.97	384
accuracy			0.99	3072
macro avg	0.99	0.99	0.99	3072
weighted avg	0.99	0.99	0.99	3072

Classification Report for CNN Model 2

	precision	recall	f1-score	support
0	0.96	0.97	0.96	96
1	0.82	0.98	0.90	48
2	0.96	0.99	0.97	96
3	0.90	0.99	0.95	96
4	0.98	0.99	0.98	96
5	1.00	0.95	0.97	96
6	0.98	0.94	0.96	48
7	0.99	0.93	0.96	96
8	1.00	0.90	0.95	96
accuracy			0.96	768
macro avg	0.95	0.96	0.96	768
weighted avg	0.96	0.96	0.96	768

Model 3.

Classification Tree on Training: CNN Model 3

	precision	recall	f1-score	support
0	0.91	1.00	0.95	384
1	1.00	1.00	1.00	192
2	1.00	1.00	1.00	384
3	1.00	1.00	1.00	384
4	0.97	1.00	0.99	384
5	1.00	0.89	0.94	384
6	1.00	1.00	1.00	192
7	1.00	1.00	1.00	384
8	1.00	0.98	0.99	384
accuracy			0.98	3072
macro avg	0.99	0.99	0.99	3072
weighted avg	0.99	0.98	0.98	3072

24/24 [==============================] - 0s 8ms/step
Classification Report for CNN Model 3

	precision	recall	f1-score	support
0	0.85	0.99	0.91	96
1	1.00	1.00	1.00	48
2	0.99	0.99	0.99	96
3	0.98	0.99	0.98	96
4	0.98	1.00	0.99	96
5	1.00	0.86	0.93	96
6	0.96	0.94	0.95	48
7	1.00	0.96	0.98	96
8	0.99	0.98	0.98	96
accuracy			0.97	768
macro avg	0.97	0.97	0.97	768
weighted avg	0.97	0.97	0.97	768

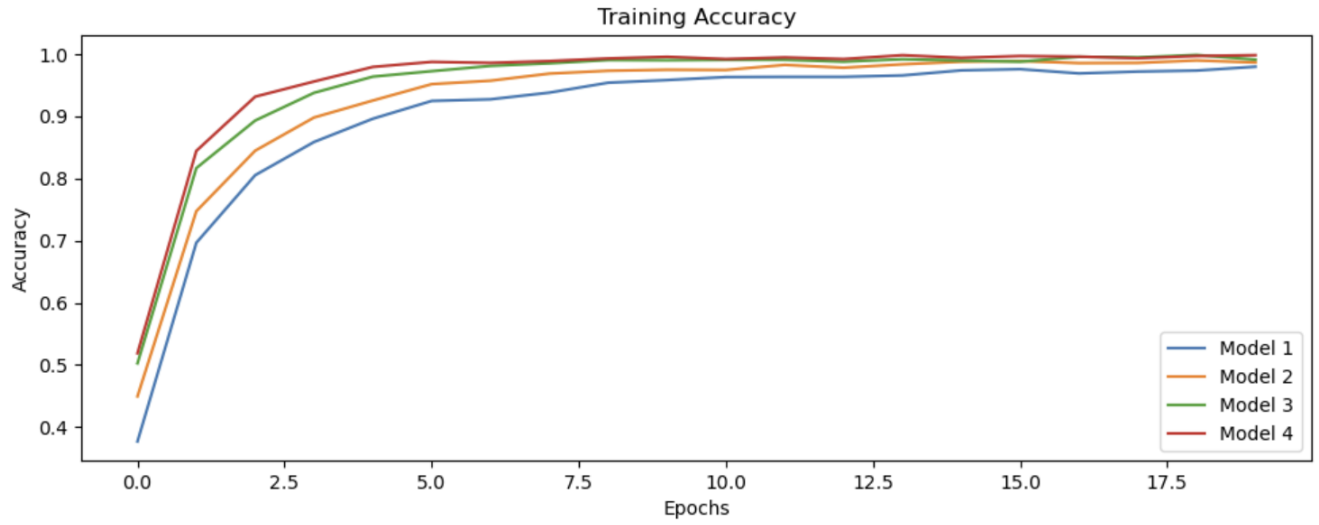
Model 4.

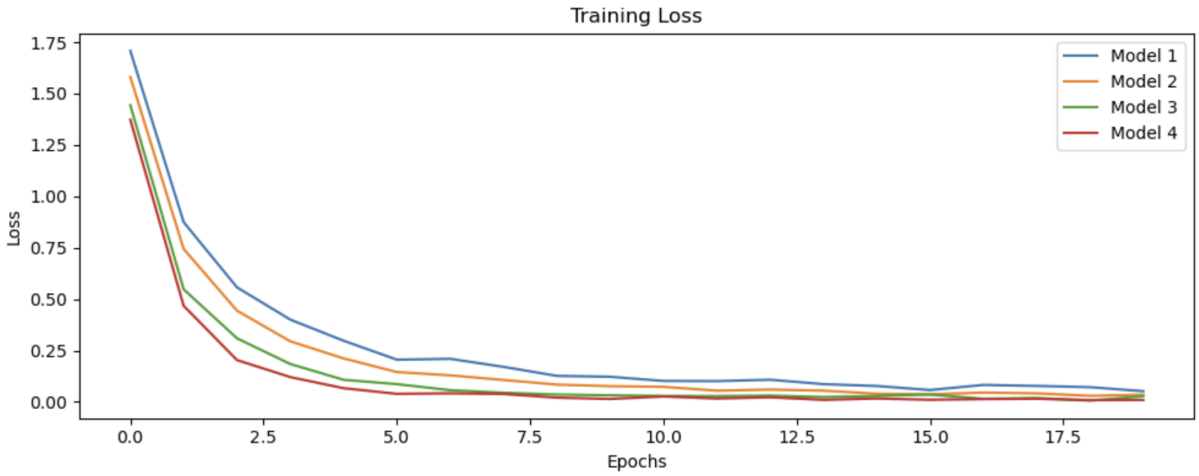
Classification Tree on Training: CNN Model 4

	precision	recall	f1-score	support
0	1.00	1.00	1.00	384
1	1.00	1.00	1.00	192
2	1.00	1.00	1.00	384
3	1.00	1.00	1.00	384
4	0.97	1.00	0.98	384
5	1.00	0.97	0.98	384
6	1.00	1.00	1.00	192
7	1.00	1.00	1.00	384
8	1.00	1.00	1.00	384
accuracy			1.00	3072
macro avg	1.00	1.00	1.00	3072
weighted avg	1.00	1.00	1.00	3072

Classification Report for CNN Model 4

	precision	recall	f1-score	support
0	0.95	0.98	0.96	96
1	0.98	1.00	0.99	48
2	0.99	0.99	0.99	96
3	0.95	0.98	0.96	96
4	0.97	1.00	0.98	96
5	1.00	0.93	0.96	96
6	0.98	0.94	0.96	48
7	0.99	0.96	0.97	96
8	0.98	1.00	0.99	96
accuracy			0.98	768
macro avg	0.98	0.97	0.97	768
weighted avg	0.98	0.98	0.98	768





We have chosen model 3 of CNN as the best model and we will be using this for final prediction of images in the class.

For the final model selection, model 3 of CNN was decided since it had the highest accuracy among other CNN models and the other algorithms which were performed during the project. After the final selection the model was trained using 100% of the data.

```
120/120 [==============================] - 1s 7ms/step
Classification Report for CNN Model on Entire Dataset
              precision    recall  f1-score   support

           0       0.99      1.00      0.99       480
           1       1.00      1.00      1.00       240
           2       1.00      1.00      1.00       480
           3       0.99      1.00      0.99       480
           4       0.99      1.00      0.99       480
           5       0.98      0.98      0.98       480
           6       1.00      0.98      0.99       240
           7       0.99      1.00      0.99       480
           8       1.00      0.97      0.99       480

    accuracy                           0.99      3840
   macro avg       0.99      0.99      0.99      3840
weighted avg       0.99      0.99      0.99      3840
```

Conclusion:

The Signature Verification Project undertaken by Team 2 aimed to develop a robust machine learning model for accurately verifying individuals' signatures. Leveraging deep learning techniques, advanced algorithms, and a substantial dataset containing 80 signature photos from each participant, the project delved into the nuances of signature dynamics, stroke patterns, and personal writing styles. The objective was to contribute to the development of authentication systems, offering a reliable tool for applications such as access control, financial transactions, and legal paperwork, while mitigating potential fraudulent activities.

The project proceeded through essential stages, beginning with meticulous Data Preprocessing. This involved standardizing image formats, resizing, cropping, and converting to grayscale to enhance the model's ability to identify authentic signatures consistently. Data Augmentation techniques were employed to enrich the dataset, and advanced Image Processing, including edge detection and thresholding, highlighted key signature features.

Model Training utilized four distinct algorithms: K-Nearest Neighbors (KNN), Random Forest, Multi-Layer Perceptron (MLP), and Convolutional Neural Network (CNN). Each algorithm underwent rigorous evaluation, considering accuracy on both training and testing data. Notably, the CNN model emerged as the most effective, achieving the highest accuracy among the models tested.

The final model, CNN Model 3, was selected for its superior performance and trained using 100% of the data. The project's success lies in its systematic approach to data preparation, algorithm selection, and model evaluation, leading to the development of a reliable signature

verification system. As a concluding step, the chosen model is poised for practical implementation, offering a valuable solution for identity verification across various domains.