FEATURE ENGINEERING PROJECT INCREMENT-1 REPORT

PROJECT: SIGNATURE DETECTION AND FORGERY DETECTION SOFTWARE.

TEAM MEMBERS:
RAM RALLABANDI {11547220}
ACHYUTANAND MISHRA {11542607}

RELATED WORK(BACKGROUND):

This project would be fed with a certain amount of data that contains a person's signature. Various documents like a person's lease, Valid driver's license and supporting legal documents that contain the user's signature would be compared to before declaring the signature would be genuine or a fraud.
We can summarize the project as comparison of two images and the output would be the similarity score between the two images. There were various projects that were done regarding this concept of similarity between the images. Some of the famous works include:

- Apple's FaceID
- Google lens by Google
- Amazon's product search by image.

These software's have the same agenda. They match the closest image that is fed through the input and compares it to the existing database. We can say that Apple's FaceID is a little bit different because it registers various facial features as the datapoints and matches them to its registered datapoints when the user tries to unlock iPhone or a MacBook.

How is the signature verification system different from the related work's:

These works have a different motive than compared to the Signature detection system. The FaceID, Google lens & Amazon projects would just compare the input to the trained dataset and return the closest found image. Clearly, there is a scope for little error.
Our project would compare the input image to various signatures of the user throughout its database. There are various features such as strokes, pressure points that would be compared by it before it declares a signature genuine or fraud. The scope for error in this project is very low. That is what makes it challenging and made us work on it.
Although we will return the similarity score in the final output, there would be a certain threshold for that score. If it is inside the threshold, it would be declared a forgery, else if it is slightly above the threshold, the software would cross check with some other signatures.

DATASET:

We will use a signature verification dataset that we got from Kaggle. The system would be trained using that dataset and after that we will pass the test image that contain's signatures of one of the team members and later we would give one genuine and one fraud signature that would be close to the genuine one and check whether the system will be able to get accurate results or not.
For now, we will use this dataset only for training purpose and not as a database of the user to compare the signatures.

DATASET LINK:
https://www.kaggle.com/datasets/robinreni/signature-verification-dataset

DETAIL DESIGN OF FEATURES:

The features we used and the detailed analysis of why the features were designed is given below:

Gray scaling:

We would grayscale the images to make sure that color of the picture would not affect the similarity scores between the images.

Noise filtering:

There are various noise filtering techniques that could be used, but since the photo would mostly be taken on mobile phones, we are leaning towards salt and pepper noise and its removal.

Edge detection:

We do not have the enough equipment to do this project on a real time signature, so we are preferring a photograph. Since we can easily see the

difference between the pen strokes using edge detection, we have decided to use it as a feature.

Features to work on:
We are still working on the *feature extractor* and the *feature descriptor* that we have mentioned in the project idea submission. We would include more about these features, their implementations and purposes in the final increment.

ANALYSIS AND IMPLEMENTATION:

Analysis:
We have worked on training our model and we have been working mostly on the accuracy. We are yet to focus on the testing part of the program. Once we finish the project completely, we are sure that there is very less amount of scope for errors. To make it almost like a real time project, we would create a profile for each signature and the signature would be matched to that specific profile and checked for forgery.

Implementation:
We are currently implementing the code in google colab and jupyter notebook, since these were the most used platforms, we are more confident in executing the code in it. Once we complete the coding part, we will upload it on GitHub and post the link for the code.

PRELIMINARY RESULTS:

We are still on the testing part of our code, because of that the system is still learning from its mistakes. For now we are able to differentiate the codes that are mostly different but, the signatures that are much alike are having less similarity score than what was normally expected. We will work on this part more keenly and try to differentiate them easily once our work on the features is completed.

IMPLEMENTATION STATUS REPORT:

Work completed:

So far, we have got a required dataset and we have added 3 features to the project. We have taken a few references from stack overflow and such to make sure we dealt with errors in a smooth way. The implementation of the idea into the code is done, we still have a little work that is left.

Responsibilities:

Ram Rallabandi:

Coding and design implementation
Working on the features' enhancement
Image Smoothening
Worked on the logical part of the source code

Achyutanand Mishra:

Data engineering and cleaning.
Training and partial testing of the available data.
Sorting/rectifying errors.
Worked on the source code for features.

Contributions:

Source code and design:
Ram Rallabandi – 65%
Achyutanand Mishra – 35%

Implementing features on the code:
Achyutanand Mishra- 70%
Ram Rallabandi- 30%

Image processing:
Ram Rallabandi- 75%
Achyutanand Mishra – 25%

Errors rectification and data processing:
Achyutanand Mishra- 70%
Ram Rallabandi- 30%

Project report(increment 1):
Ram Rallabandi -50% Achyutanand Mishra- 50%

WORK TO BE COMPLETED:

Description:
We would still work on the errors where the close signatures are having Comparatively less similarity score than to be expected.
There a still a couple of features to be implemented. The coding part is done, there are errors that must be dealt with.
Concerns:
We are having a little bit of trouble with the similarity score.
The errors with the coding part are remaining and they will be dealt with soon.

GITHUB LINK:

https://github.com/Achyut2995/SignatureVerification