**SIGNATURE RECOGNITION MODEL**

**PROJECT TITLE:**

Build a Neural Network based solution to distinguish between real and forged signatures of a particular person.

**ABSTRACT:**

The target of this project is to use neural networks to recognize the **real and forged signatures** of a person. The network that I made here can be used to classify the real and forged signatures of any random person whose signature was not even considered for training the network.

Signature Recognition task is of utmost importance from the point of view of security , law enforcement, and prevention of misuse of a person’s identity by using his/her .

**Keywords:** Writer-Independent, Writer-Dependent, Face Recognition, Siamese Network, CNN, VGG16, Dense Neural network, Sigmoid.

**INTRODUCTION**

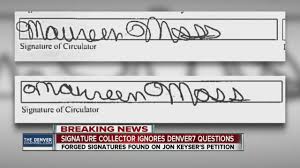
Signature Recognition task is of utmost importance from the point of view of security, law enforcement, and prevention of misuse of a person’s identity by using his/her. Now a days the digital signatures are quite common, people paste images of their signatures on online documents, and that was what I had done while accepting the offer for this internship. In this scenario it has become quite important to be able to verify the genuinity of the signature.

**SCIENCE BEHIND USE OF SIGNATURE AS IDENTITY MARKER:**

The Phrase that ***“each person is different in its own way”,*** is actually biologically correct. Each human body carries with itself a unique identification marker, which is unique to that body only. We have several biological markers, like, biometric fingerprint, DNA, etc as well as several behavioural markers like writing style, voice, and **signature** etc. There is a proper reason why signature is unique identity marker because a specific part of the brain is involved in learning to write a signature. Its kind of a training for the brain to to replicate a specific sequence of motor movements having a person specific patterns, like relative gaps, relative height and width, tilt, etc.

This ensures that a forged signature will certainly be different from the original one as probability that the person forging the signature will learn the same motor sequence of the original person is very very low especially when he/she has only access to the signature of the original person and not the video of person doing the signature. That’s why the signatures are quite unique identifiers for recognizing people. Signature is very much a subconscious activity, and trying to reproduce it consciously will certainly lead to markers distinguishing it from real.

The following image show real(upper) and forged(lower)



To know more about the forensic Science behind this, you can visit the link in references section at the end of the report.

Several attempts have been made by the forensic experts and scientists to develop method to distinguish forged from real signature. The traditional approaches mainly involved extracting the image features and then comparing them to image features of the real signature. Following are some of the Hand Engineered features :

**Geometric Features**: Geometric features measure the overall shape of a signature.

**Graphometric features**: this includes Calibre - the ratio of Height / Width of the image; Proportion, referring to the symmetry of the signature, Alignment to baseline - describing the angular displacement to an horizontal baseline,

and Spacing - describing empty spaces between strokes.

**Directional features**: Directional features seek to describe the image in terms of the direction of the strokes in the signature.

**Mathematical transformations**: Researchers have used a variety of mathematical transformations as feature extractors. Hadamart transform and spectrum analysis, Contourlet transform, discrete Radon transform, etc are use as feature extractors.

**Shadow-code**: It is an Extended Shadow Code for signature verification. A grid is overlaid on top of the signature image, containing horizontal, vertical and diagonal bars, each bar containing a fixed number of bins. Each pixel of the signature image is then projected to its closest bar in each direction, activating the respective bin. The count of active bins in the projections is then used as a descriptor of the signature.

But recently the growing power of Neural networks and their success in image analysis has promised a new door to look into this problem. Several attempts have been made by Machine Learning community in this regard. Following are some examples of such efforts:

1. RBMs:

Ribeiro used RBMs to learn a representation for signatures, but only reported a visual representation of the learned weights, and not the results of using such features to discriminate between genuine signatures and forgeries.

1. Feature Extractor Training Approach(Using CNNs)

Hafemann proposed a **Writer-dependent** feature learning method, where a the data is used to learn a feature representation f(X). This representation is learned using a Convolutional Neural Network (CNN) . Then this trained extractor is used to compare feature vectors of one image against the real one and then it is determined weather they are same or not. This method is **writer dependent** meaning it can be trained for only one person’s signature.

Also there have been several attempts like using SVM’s , and HMM’s and ensemble learning models. But the shear complexity of the task is huge and is still a hot topic of research today.

Reason why signature classification is hard :

The signature of same person varies hugely, this is because a lot of their external factors like inclination of hand, thickness of pen , etc can also affect the signature. The following is the image showing several real signatures of same person, on top each other.



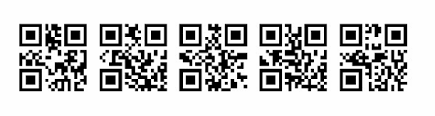
If the signature is forged by a skilled person, then the difference between real and original are almost nil to a human eye. Which is quite evident to emphasize that it is not same as classifying a image.

Hence the High Intra-variability and High Inter-similarity are the reason that task of distinguishing between forged and real signature is difficult.

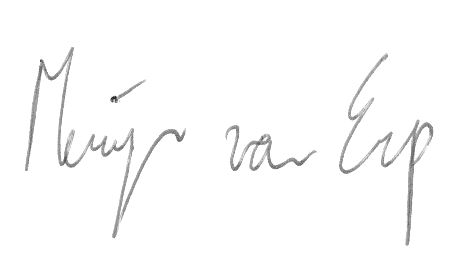
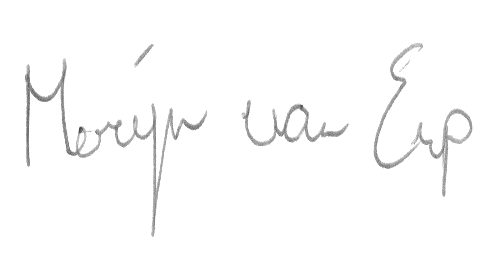
**PROBLEM STATEMENT**

To build and train a Neural network model for taking in the following input [image of real signature of a person, image of a signature of the same person], and output a number in range [0,1] marking ***how different*** are the two images. So, 0 would mean that two images are similar and hence the second signature is real, and 1 would mean the opposite.

**EXPECTED OUTPUT**



**INPUT**

**(Real) (To be tested)**

**EXPECTED OUT**

Number in range [0,1] marking how **different** tobetested is from real. If less than 0.5, we say that tobetested is real signature else, it is forged.

# IMPLEMENTATION AND UNIT TESTING METHODOLOGY

**DATA:**

I have checked on the net and did not find any large dataset real and forged signatures. All large datasets require academic verification to access them. I found a small dataset on Kaggle. The link for the same is as follows:

<https://drive.google.com/drive/folders/1R9uQ1bj4MpL0i_HtKX3KVSlBu3gK56_P?usp=sharing>

This contains real and forged images of 12 participants, which gave their signatures as well as forged each other’s signature. The images are named in the following manner:

NFI-00602023 is an image of signature of person number 023 done by person 006. This is a forged signature. NFI-02103021 is an image of signature of person number 021 done by person 021. This is a genuine signature.

**MODEL BUILDING:**

I have built and trained the model in Google Colaboratory, an online free Cloud Computing platform from Google, having Support for GPU and TPU and is quite suitable for creating and training small and moderate sized neural networks. The link for the same is as follows, It is advised to open the following link and follow the code along with upcoming explaination.

<https://colab.research.google.com/drive/17Clgjs7lOpuWTHDv5m9OgF5yj-EDYzXu?usp=sharing>

**ARCHITECTURE:**

The model consists of two parts namely, encoder and comparator

ENCODER : takes three images, runns CNN over all of them and finally outputs three 500 dimensional vector for each image, encoding the image features. Loss function aims to reduce the gap between outputs of real signatures pair , and widen the gap between vectors of real and fake image pair. It is trained on the data generated as follows.

COMPARATOR : This takes two images, one of real signature, and other of the signature which is to be tested against it. Thiese images arre passed through the encoder above, and their freatures are extracted. These are then passed through another neural network of densely connected layers which outputs a number between 0 and 1 signifying closeness. i.e. the more close to 0, the more is the second signature is matching to the real one.

This is trained by keeping the encoder weights fixed and training the densely connected neural- network ahead of it.

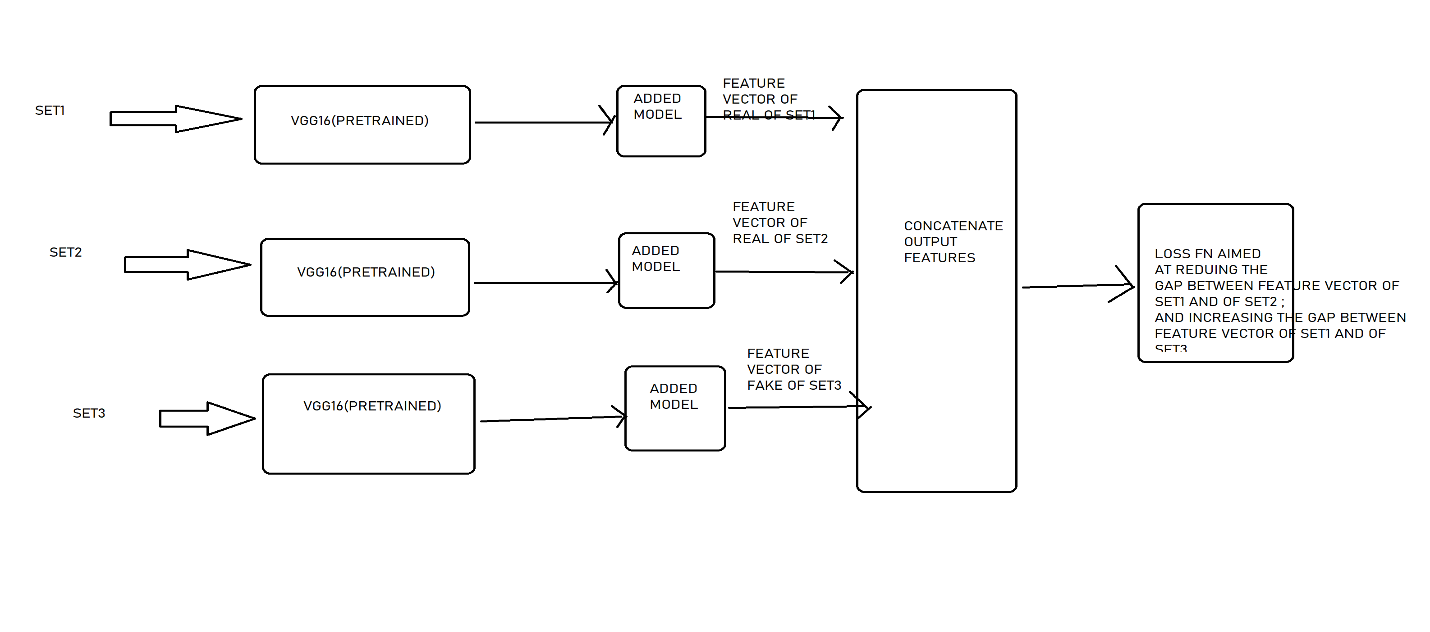
*.\_. I tried to train the whole network at once, but it did not work out well. The reason might be that the weights of the ENCODER, which are trained using Approach for Face Recognition, are trained to differentiate between the images. But training them from comparatoe point of view, might force it to leave this behaviour and assume some other behaviour which is not able to differentiate between very similar images(as all fake signatures are).*

**ENCODER MODEL:**

Here we are going to use the Architecture used for Face Recognition model. The Network will take a set of three images (real signature of a person, another real signature of same person, fake signature of that person), and output three encoded vectors of length 500 each, encoding the features of each image.

The good thing about the mode is that it is not writer specific, i.e., once it is trained it can differentiate between real and fake signatures of any other person who might not be in the training set.

on the other hand, the downside of the model is that it requires fake signatures of persons for training, which are less practical way of approaching the problem.

****

The steps involved in building and training the encoder are as follows:

1. Import Required Libraries. Namely cv2 (OpenCV),keras, tensorflow, numpy, os, and random
2. Prepare the data. We resize the images to size(50, 125, 3). And store different categories of images in different variables as shown below.

real001 -->> list of Image of real signatures of person 001

real002 -->> list of Image of real signatures of person 002

.

.

.

real005 -->> list of Image of real signatures of person 005

Fake001 -->> list of Image of fake signatures of person 001

'

'

'

'

fake005 -->> list of Image of fake signatures of person 005

1. We prepare the training set by making triplets of images in the following way

(real signature of a person, another real signature of a person, forged signature of the same person)

Set1 contains the first type, set2 the second and set3 the third type of images.

1. Next we load the pretrained VGG16, by excluding its last prediction output layer.
2. Next we build a model named added\_model containing a dense layer of size 500,

The size of our feature vectors.

1. Then using the above two models, we construct the ENCODER MODEL as shown in the picture above. Note here that the weights of VGG16 and added model are shared here for set1, set2, set3 inputs.
2. We construct a loss function which when reduced will cause the distance between set1 and set2 feature vectors to reduce and between set1 and set3 feature vectors to increase. For this we use

Loss = sigmoid(mse(feature of set1, feature of set2) – mse(feature of set1, feature of set3))

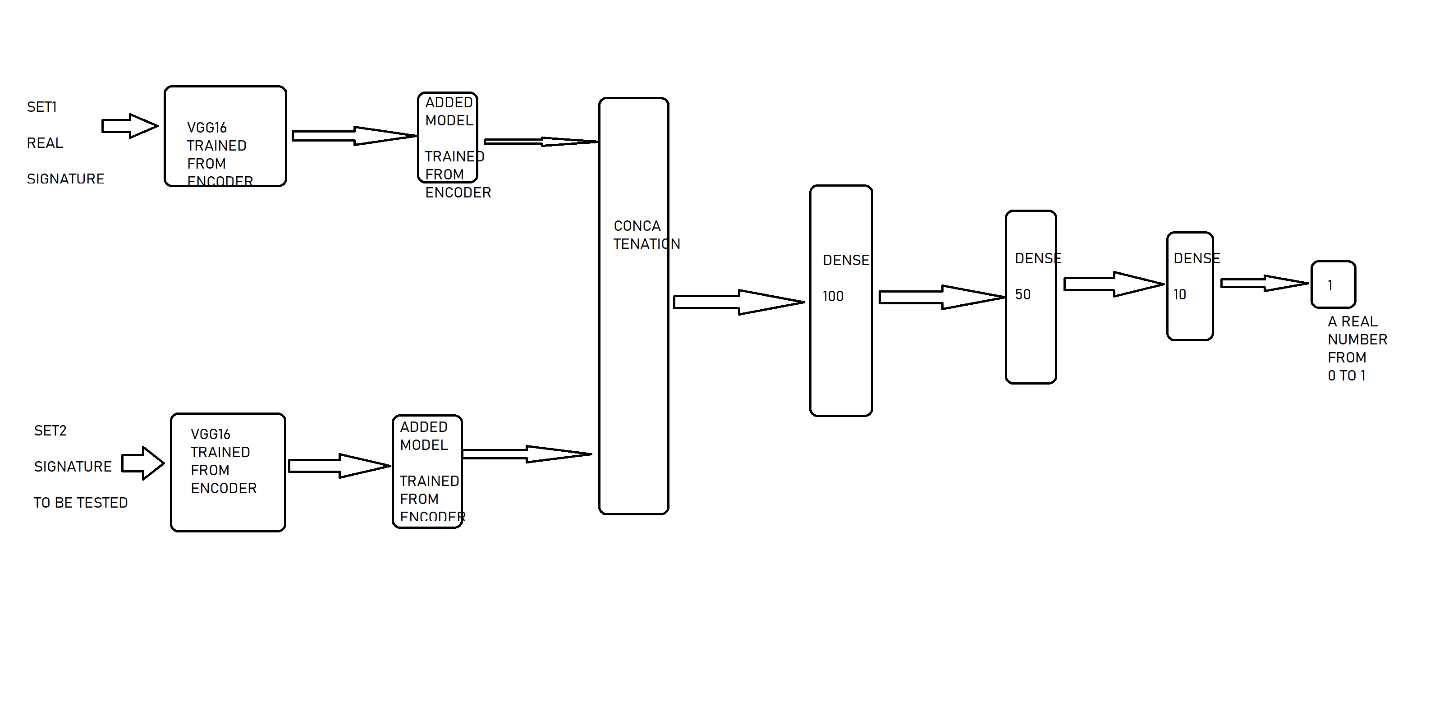
**The sigmoid is calculated to prevent overflow of tensor values as it maps the values in interval 0 to 1, hence reducing the chances of overflow.**

1. We compile it using Adam optimizer.
2. Then we train the model.
3. Note that the VGG16 and added\_model models were also trained here(as they are parts of the encoder model), so we can use them to get the feature vector of an input image .

**COMPAROR MODEL:**

This model takes two images (first the real signature of a person, second the one to be tested against the first one for real or fake), and outputs a real values between 0 and 1 depicting the closeness of the second image to the first one in sense of closeness of signatures, i.e. the more close the value is to 0 , the more the probability that the second and first signatures are the real signatures of same person else if value is close to 1 , then the second signature is forged version of the first signature.

Note: the Comparator uses the VGG16 and added\_model models trained in the above step of ENCODER model training, which are used to extract the distinguishing information from the signature images.



The steps involved in building and training the comparator are as follows:

1. Data Preparation : Prepare the data in the following format (real signature of a person, real/forged signature of that person), and the corresponding element of Y array will have 1 if second image is real , else it will have -1 as value. Y will ve used for designing the loss function. Set1 contains first images, set2 contains second images.
2. Construct the comparator model on the lines of as shown in the above image.
3. Make VGG16 and Added Model models as untrainable.
4. Create the loss function which will aim to reduce the output from (real, real ) type inputs and increase the output from(real, forged) type inputs.

This is achieved by multiplying Y with the comparator output elementwise and taking the sum.

1. After compiling, train and save the model.

**Note: performing step 3 is of utmost importance as I have tried to train the whole model as whole and the results were not pretty.**

**TESTING**

We test the comparator by creating the test set similar to the one made for training.

We get an accuracy of 100 percent accuracy on both test and train set. This is due to the test and train sets are soo less in size.

The outputs can be viewed on the Colaboratory notebook which I have shared.

**CONCLUSION**

The conclusion is that using Face Recognition approach to train a Feature encoder for signature images and then using that trained feature generator to train another network for predicting the closeness of the two signatures can infact lead to high performance signature recognition model, which is capable of differentiating the signature of people whose signatures were not there for training the network, i.e. it is writer Independent.

**REFERENCES**

Science behind uniqueness of Signatures - <https://books.google.co.in/books?id=y_abDwAAQBAJ&pg=PA45&lpg=PA45&dq=area+in+brain+dedicated+to+handwritten+signature&source=bl&ots=Im6UBR31XS&sig=ACfU3U3K09k5HsEkSnMDUOfrLeRkgY6d2w&hl=en&sa=X&ved=2ahUKEwjC3crG-ZnqAhVmyDgGHT7iAcEQ6AEwCnoECAkQAQ#v=onepage&q=area%20in%20brain%20dedicated%20to%20handwritten%20signature&f=false>About Bar codes - <https://www.youtube.com/watch?v=XPuTZMp-HE8>

Paper related to current status of Signature Recognition through ML- <https://www.researchgate.net/publication/320309808_Offline_Handwritten_Signature_Verification-Literature_Review>

An attempt to Create signature Recognizer - <https://www.ijrte.org/wp-content/uploads/papers/v7i6s/F02900376S19.pdf>

Working with Google Colab :

<https://towardsdatascience.com/getting-started-with-google-colab-f2fff97f594c#:~:text=You%20can%20create%20notebooks%20in,upload%20Kaggle%20files%2C%20download%20your>