Ghost Fingerprint Detection

Aishwarya Mala GM
EEMCS, University of Twente,
S2242842
a.m.gurusamymuthuvelrabindran@student.utwente.nl

Christopher Benjamin Leonard
EEMCS, University of Twente,
S2378515
c.benjaminleonard@student.utwente.nl

Navneeth Paul
EEMCS, University of Twente,
S2273756
navneetpaul@student.utwente.nl

***Abstract* – A system to detect the presence of ghost prints in an input fingerprint image has been proposed. The algorithm consists of three main subdivisions – Preprocessing, Feature Extraction and Classification. After the basic preprocessing steps, the fingerprint image is divided into image blocks. Features that are based on the local orientation, frequency and mean pixel intensity are extracted from each of the image blocks. The local orientation is calculated using the Sobel operator. Then the local ridge frequency is calculated based on the location orientation obtained. Fourier Transform is used to obtain the first and the second prominent frequencies and orientations. The difference in these prominent frequencies and orientations is used as a feature. The mean intensity of all the pixels that represent the ridges in the input fingerprint is also considered as a feature. The final step was to train classification models (Neural Networks, Random Forest and K-nearest Neighbors) on the features extracted to detect the presence of ghost prints in the input.**

Keywords—*Ghost Fingerprints, Ridge Frequency and Orientation, Fast Fourier Transform, Machine Learning Classification.*

I. Introduction

Personal identification systems are one of the most crucial aspects of authentication and are being used to restrict access to high-level security systems and personal information. Use of physical characteristics that are unique to a person and easily collectable like biometrics are considered as a safer option when compared to the conventional approaches of using passwords or ID cards [1, 2]. Fingerprints, irises, face, voice etc, are some of the most commonly used biometrics. These characteristics are unique to each person and thus provide the means for more secure identification systems.

When we consider a biometric identification system that uses fingerprint images as the main biometric characteristic for identifying an individual, it is crucial to ensure that the fingerprint images in the system are free of any contaminations. In this work, we have worked with a kind of contamination that is often referred to as “ghost fingerprints”. These contaminations could adversely affect the performance and efficiency of the identification system. Ghost fingerprint occurs when a second unwanted fingerprint overlaps part (or even whole) of the primary or wanted fingerprint [2]. Some of the common reasons for the occurrence of ghost fingerprints could be attributed to the presence of fingerprints left behind on a surface and fingerprint scanners that are not properly cleaned [4]. Fig.1. depicts an example of a ghost fingerprint.

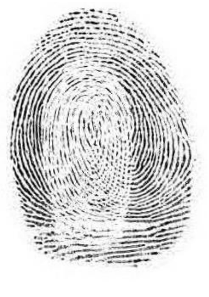


Fig. 1. Example of a Ghost Fingerprint

David Maltoni in “Handbook of fingerprint recognition”, describes that a commonly used fingerprint recognition system could be subdivided into three classes or methods: Non-Minutiae feature-based matching, Minutiae-based matching and Correlation-based matching [3]. All these proposed approaches use either the ridge shape, frequency characteristics, minutiae locations etc, or even a combination of these characteristics. Since all of these groups are affected by the presence of ghost fingerprints, the efficiency of the recognition system will be adversely affected while matching the fingerprints. It is therefore very much essential to detect the presence of ghost fingerprints prior to loading them in the database or used by biometric recognition systems.

II. Related Works

A literature survey was conducted on the already existing work with regard to overlapping (ghost) fingerprint separation and detection.

Orczyk, T., et al [6] investigated the fingerprint ridge distance and developed an efficient means to estimate the values of local ridge frequency. Ridges frequency, as a global feature of fingerprint, is very important to image preprocessing methods used in automatic fingerprint identification systems (AFIS). They proposed an algorithm, through the use of fingerprint orientation in the image, where the frequency values are calculated directly, without the use of computationally expensive spectral analysis. This approach proved to be effective in the comparison with the previously adopted approach by X. Zhan, et al [7] where the ridges frequency by means of the Fourier Transform and the radial distribution functions are computed.

Huang, H. C., et al [5] developed a system that can automatically and precisely separate an overlapping fingerprint into two areas, i.e., an overlapping area and a non-overlapping area, by analyzing their orientation fields and complicities. The approach deals with a recursive correction algorithm and a restrictions relaxation labelling algorithm applied to separate two fingerprints from the determined overlapping fingerprint area. Later Gabor filter was applied to enhance the figure quality of the two separated fingerprints. Receiver Operating Characteristic (ROC) and Cumulative Match Characteristic (CMC) were utilised to examine the performance of the proposed system and it was seen to achieve 85.7 % accuracy in recognition rate.

Chen F., et al [8] proposed a similar approach of separating overlapped fingerprints into individual fingerprint components. The first step involved using a local Fourier transform to estimate an initial overlapped orientation field, which contains at most two candidate orientations at each location. Then relaxation labelling technique was employed to label each candidate orientation as one of two classes. Based on the labelling result, the initial overlapped orientation field was separated into two orientation fields. Finally, the two fingerprints were obtained by enhancing the overlapped fingerprint using Gabor filters tuned to these two components separated orientation fields, respectively. Satisfactory results were obtained on latent overlapped fingerprints.

With this literature as the base, we have proposed a system that could detect the presence of ghost prints in a given fingerprint image.

III. Dataset

Real fingerprint image dataset provided by Dutch police has been used for training and testing of the developed system. The fingerprint images in the given dataset were labelled into 7 classes - class 0 to 6. Class 0 consists of fingerprints that are uncorrupted by any ghost fingerprints or any other contaminations. Fingerprint images that were highly affected by ghost prints were labelled as class 1. Classes 2-5 included proportionally less affected fingerprints with the least affected fingerprints labelled as class 5. Fingerprints that were contaminated by other than ghost fingerprints were labelled as class 6. There are a total of 4650 RGB images. Not all the images are of the same size and the images are not distributed equally within the classes.

IV. Method

A method was developed to detect the presence of ghost fingerprint based on the underlying principle that the regions that are corrupted by a ghost fingerprint will show a significant difference in ridge frequency, orientation and pixel intensities when compared with the uncorrupted areas.

The proposed methodology contains three main subdivisions: Image pre-processing, Feature extraction and Classification. Each subdivision is explained below.

A. *Pre-processing*

Before extracting the features it was necessary to do a few pre-processing steps. The fingerprint images in the dataset were RGB images with a white background. For the proposed analysis only grayscale and binary versions of the image were required. Thus as the first preprocessing step, the input RGB image was converted into a grayscale image that had intensity values in the range [0 1].

Before thresholding the images and obtaining their binary versions, we had to address the issue of non-uniform image sizes in the dataset. Resizing or scaling the images might lead to loss in information so a decision was made to pad 1's analogous to the white backgrounds of the fingerprints. All the images were brought to a uniform size of 768x800. This final image size was arrived upon by taking into consideration the largest image dimension in the dataset and the fact that during the feature extraction steps the input image will be split into blocks of size 32x32.

The next step was to binarize the input images. Otsu's method was used to compute the threshold value. Otsu's method aims at computing the optimum threshold value that maximizes the inter-class variance between the foreground and the background pixel intensities. This threshold was calculated using the MATLAB function *graythresh*. This threshold was applied to the grayscale image to obtain the binary image. The binary image is then morphologically opened and closed to remove unwanted discontinuities and noise.

The grayscale and binary images are then divided into blocks of size 32x32 for extracting features. Fig. 2. shows a flowchart with the pre-processing steps.

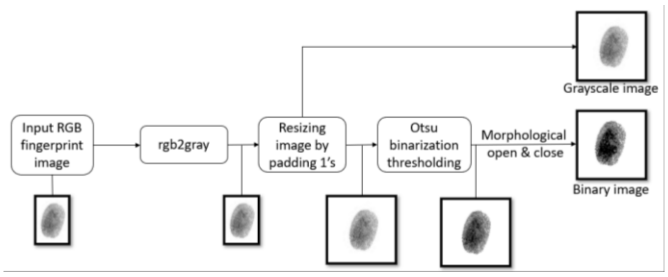

Input RGB fingerprint image
rgb2gray
Resizing image by padding 1's
Otsu binarization thresholding
Morphological open & close
Grayscale image
Binary image

Fig. 2. Pre-processing steps

B. *Feature Extraction*

Presence of ghost fingerprint is mainly characterized by the changes in ridge frequency, orientation and pixel intensity values. The intensity values of the pixels that form the ridges are directly obtained from the grayscale image. Two of the most common methods that were found in the literature for calculating the ridge frequency and orientation were taken into consideration – Using gradient operators (like Sobel) [6] and Fast Fourier Transform (FFT) [8]. The preprocessed grayscale and binary versions of the input image are divided into image blocks of size 32x32. The minimum of the pixel values of each of the binary image blocks is calculated. If this minimum value is 1, then it implies that the block completely lies in the image background and these blocks are neglected. Each of the non-background binary image block whose minimum pixel value is 0 and its corresponding grayscale image block are further processed for extracting features.

a. Ridge Frequency and Orientation – Sobel Operator

Sobel is a discrete differentiation operator and it is usually used to emphasis the edges in the given image. The operator consists of two 3x3 kernels, S_x and S_y (Given in equation 1). S_x kernel on two-dimensional convolution with the input image gives the approximate derivative value in the horizontal direction and similarly, the kernel S_y will give the approximate derivative values in the vertical direction. In equation 2, I represents the input image, I_x and I_y represent the horizontal and vertical derivatives of the input image I respectively, and * represent 2-dimensional convolution. A MATLAB function was developed to take an image as input and return the horizontal and vertical derivatives of the input image by applying the Sobel operator.

$$\mathrm{S_x} = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \text{ and } \mathrm{S_y} = \begin{bmatrix} +1 & +2 & -1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \tag{1}$$

$$\mathrm{I_x} = \mathrm{S_x} * \mathrm{I} \text{ and } \mathrm{I_y} = \mathrm{S_y} * \mathrm{I} \tag{2}$$

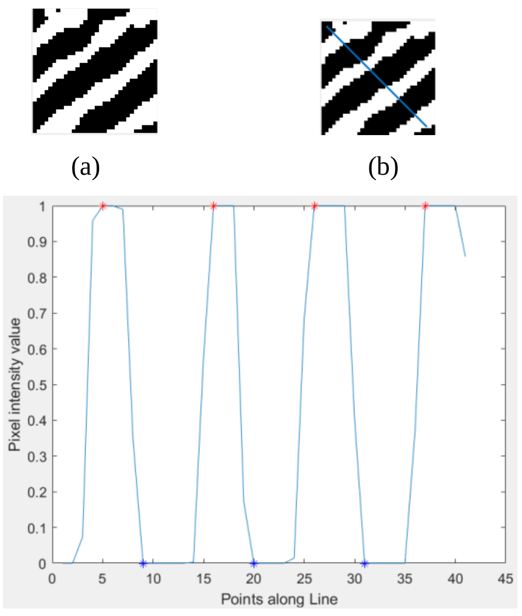
The local orientation θ of the block of size $w \times w$ is calculated as per equations (3-5).

$$\vartheta_x = \sum_{u=1}^{w}\sum_{v=1}^{w} 2I_x(u,v)I_y(u,v) \tag{3}$$

$$\vartheta_y = \sum_{u=1}^{w}\sum_{v=1}^{w} I_x^2(u,v) - I_y^2(u,v) \tag{4}$$

$$\theta = \frac{1}{2}\, tan^{-1}\left(\frac{\vartheta_x}{\vartheta_y}\right) \tag{5}$$

Using the local orientation θ obtained, the local frequency is calculated. Considering the center pixel of the block as the origin, a line is constructed at an inclination of θ. The local maximums along this line will correspond to valleys and the local minimums along this line will correspond to ridges. Fig.2 shows a sample block, the line drawn and a plot of the intensity values that correspond to the selected points along the line. The number of local maximums is taken as the number of valleys in the block and the number of local minimums is taken as the number of ridges in the block. The frequency is calculated as the mean of the number of ridges and the number of valleys in the block.



(c)

Fig. 3. Calculating Local Ridge Frequency. (a) A sample image block. (b) The image block with the line drawn by considering the block center as the origin and at an angle of θ (local orientation). (c) Intensity plot of the pixels along the line. The local maximums in this plot correspond to valleys and the local minimums correspond to ridges of the fingerprint in the image block.

The plot in Fig. 3(c) is the intensity plot corresponding to the line and image block shown in Fig. 3(b) and Fig. 3(a). The red points in Fig. 3(c) correspond to local maximums (valleys) and the blue points correspond to the local maximums (ridges). The frequency f of the image block is calculated as per equation (6) given below.

$$f = \frac{N_r + N_v}{2} \tag{6}$$

N_r is the number of ridges which is nothing but the number of local minimums and N_v is the number of valleys which is nothing but the number of local maximums. For the sample image block shown in Fig. 3, $N_r = 3$ and $N_v = 4$. Thus, the local ridge frequency of the block will be 3.5 ridges/block. A MATLAB function to calculate the orientation of an input image block was written. Then another MATLAB function that takes as input the orientation calculated was written and was used to calculate the frequency of the input image block.

b. Ridge Frequency and Orientation – Fourier Transform

Fourier Transform of an image can be used to obtain the prominent frequencies and orientations. Ideally, in regions that are corrupted by ghost fingerprints, the first prominent frequency obtained from its Fourier Transform must correspond to ridge frequency of one of the fingerprints and the second prominent frequency must correspond to the ridge frequency of the other fingerprint. In the absence of any ghost fingerprints, ideally, the first prominent frequency must correspond to the local ridge frequency of the fingerprint and the second prominent frequency must be a harmonic of the first prominent frequency.

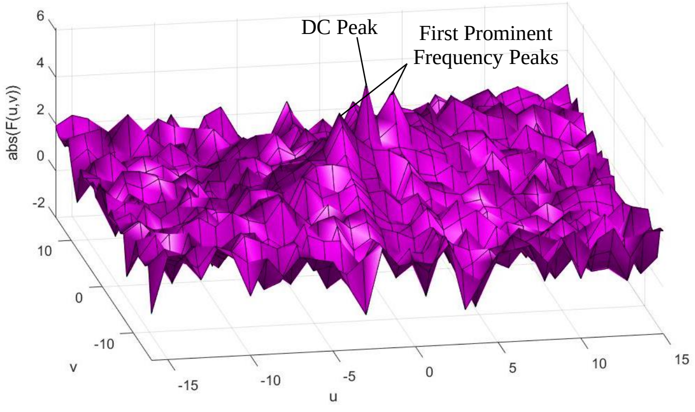


Fig. 4. A 3D plot of Fourier Transform of a sample image block showing the DC peak and the peaks that correspond to the first prominent frequency.

The Fourier Transform of the image block was computed by using the MATLAB function *fft2* and by shifting the output using the MATLAB function *fftshift*. After the shift, DC peak will be at the center at u=0 and v=0. The maximum that is found after neglecting the DC peak will be used to compute the first prominent frequency and orientation. To calculate the second prominent frequency, the two peaks that correspond to the first prominent frequency is neglected along with the DC peak. The maximum thus found will correspond to the second prominent frequency and orientation. The frequency and the orientation are calculated from the u and v by using the equations (7) and (8).

$$frequency = \sqrt{u^2 + v^2} \tag{7}$$

$$orientation = tan^{-1}\left(\frac{v}{u}\right) \tag{8}$$

A MATLAB function was written to calculate the first and the second prominent frequencies and orientations of the input image block. The function will return the differences between the first and the second prominent frequencies and orientations. The function will also check if the second prominent frequency is a harmonic of the first or not and accordingly return a frequency difference.

c. Ridge Pixel Intensity

Fig. 5 shows a fingerprint image that has been corrupted by a ghost fingerprint. It could be seen that there is a significant difference between the intensity range of the ridge pixels in the ghost fingerprint region and the intensity range of the ridge pixels in the uncorrupted fingerprint region. Thus the intensities of the pixels that represented the ridge of the fingerprint can help detect the presence of ghost fingerprint. A MATLAB function was developed to calculate the mean intensity of all the pixels that belong to the ridge regions of the fingerprint in a given input image block.

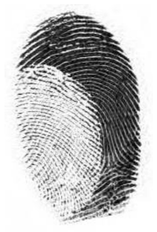


Fig. 5. A sample fingerprint image that has been corrupted by ghost fingerprint. This sample shows a significant difference in pixel intensities between the corrupted and uncorrupted regions.

d. Creation of the Feature Array

A feature array of dimension 1x12 was created for each image in the dataset. At this step of the algorithm, we will have five arrays each of size 24x25. The size of the arrays corresponds to the number of blocks the preprocessed image was divided into. Each array element corresponds to a feature extracted from an image block. The array elements that correspond to background image blocks are all set to zero in all the five arrays. The five arrays correspond to the five local features that were extracted at each image block – ridge frequency and orientation obtained using the Sobel operator, the difference between the first and second prominent frequencies and orientations obtained from Fourier Transform and mean ridge pixel intensity.

The mean and standard deviation of the non-zero elements of the above mentioned five arrays form the first 10 elements in the feature array. Presence of ghost fingerprints will also give rise to sudden changes in local ridge frequencies and orientations. Thus the normalized number of sudden peaks present in the frequency and orientation arrays were used as the final two elements in the feature array.

MATLAB code was developed to create the feature array for each image in the dataset. This feature array along with its label was then used to train classification models to detect the presence of ghost fingerprints.

C. Classification

As already mentioned, the image dataset contains class 0 with uncontaminated fingerprint images and class 1 to 5 with fingerprint images contaminated by ghost prints. The fingerprints that are expected to be most affected are in class 1, with classes up to 5 being affected proportionally less. We suspected that the proportion to which a fingerprint is corrupted by a ghost will affect the probability of it being classified as an uncorrupted or corrupted fingerprint. To study this effect 5 different datasets were created and were used to train and test the classification models and the results of each dataset were recorded separately.

Each dataset contained 681 (a few images were filtered out from class 0 as improper) feature arrays that were collected from class 0 and labelled as '0'. For dataset 1, feature arrays that were collected from class 1 were labelled '1' and were added to the existing records from class 0. For dataset 2, feature arrays that were collected from class 1 and 2 were labelled as '1' and were added to the existing records. Similarly, dataset 3 was created from feature arrays collected from 0,1, 2 and 3; dataset 4 from feature arrays collected from 0, 1, 2, 3 and 4; dataset 5 from feature arrays collected from 0, 1, 2, 3, 4 and 5. The datasets were kept balanced by limiting the number of records labelled '1' by random selection. The ratio between the number of records labelled '0' and the number of records labelled '1' was kept close to 1:1.

We trained three classification models – Neural Networks, Random Forrest and K-nearest Neighbor. Each dataset was further divided into three – training (80%), validation (10%) and testing data (10%). Training data was used to train the classification model. The performance of the model on the validation data was used to tune the hyperparameters of the model and the testing data was used to evaluate the final model performance. The entire classification segment of the project was done on Python using sklearn library [9].

a. Neural Networks

A simple neural network model was trained with the default learning rate of 0.001 and L2 regularization parameter of 10^{-5}. Using 'relu' as activation function yielded the best performance on the validation data when compared to the other activation methods provided by the sklearn library. The default weights optimization algorithm 'adam' was used and the value for maximum iterations was set to 40000. A hidden layer of size 80 seemed ideal based on the performance on the validation data from the five datasets. Addition of more hidden layers did not have a significant positive effect on the performance and thus it was stopped with one hidden layer. Fig. 6 shows the plot of accuracy on training and validation data from dataset 1 with the change in hidden layer size. The training of the model on other datasets is summarized in table 1. It gives the maximum accuracy that was obtained on the validation data.

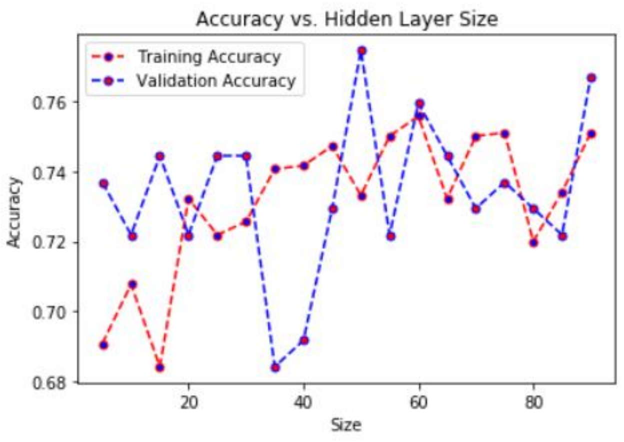


Fig. 6. Training of the Neural Network model on dataset 1 – Plot of training and validation accuracy with change in hidden layer size.

TABLE 1. Summary of Neural Network model training on the datasets

Dataset	***Validation Accuracy***
Dataset 1	0.7744
Dataset 2	0.6917
Dataset 3	0.6940
Dataset 4	0.6940
Dataset 5	0.7143

b. Random Forest

Random forest models with the number of estimators in the range [5,50] were trained on all the five datasets. Fig. 8. Shows the plot of the accuracy obtained on the validation data of dataset 1 against the number of estimators. A model with 25 estimators seemed ideal after the results that we obtained

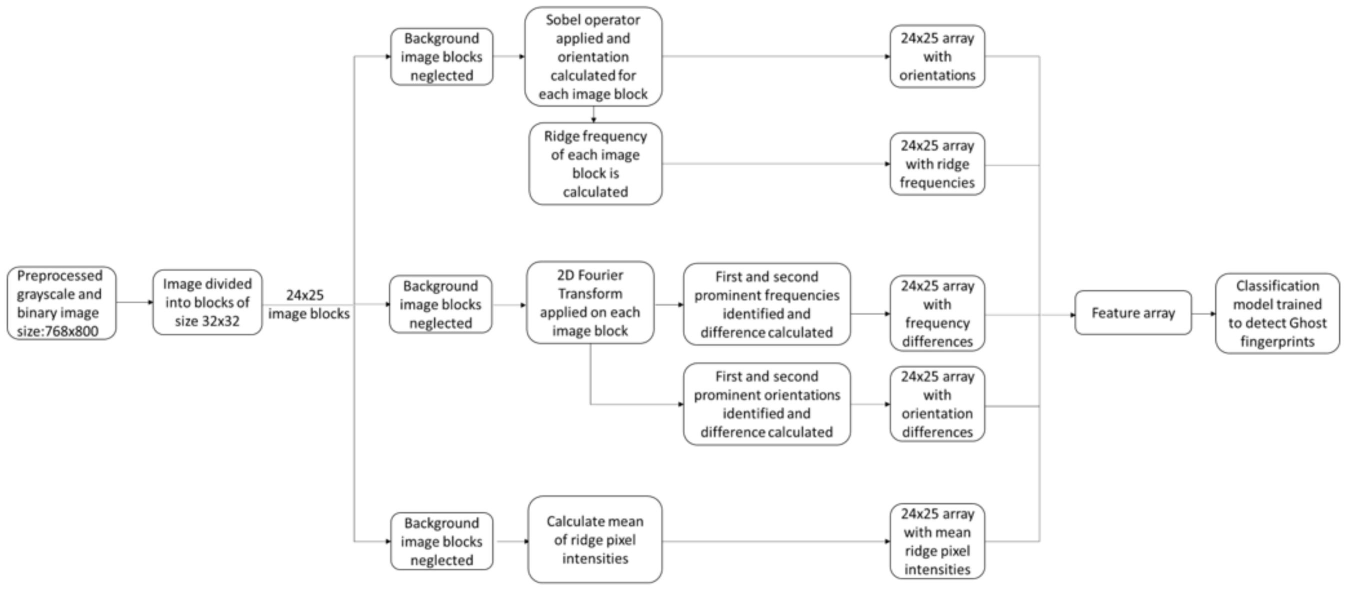

Preprocessed grayscale and binary image size:768x800
Image divided into blocks of size 32x32
24x25 image blocks
Background image blocks neglected
Sobel operator applied and orientation calculated for each image block
Ridge frequency of each image block is calculated
24x25 array with orientations
24x25 array with ridge frequencies
Background image blocks neglected
2D Fourier Transform applied on each image block
First and second prominent frequencies identified and difference calculated
First and second prominent orientations identified and difference calculated
24x25 array with frequency differences
24x25 array with orientation differences
Background image blocks neglected
Calculate mean of ridge pixel intensities
24x25 array with mean ridge pixel intensities
Feature array
Classification model trained to detect Ghost fingerprints

Fig. 7. A flowchart representing the steps that follow after pre-processing

on the five datasets. The highest accuracy that was obtained on the validation data of each dataset during the training of the Random Forest model is summarized in Table 2.

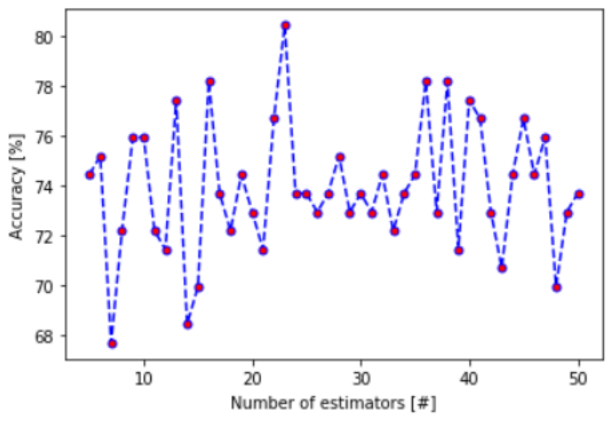


Fig. 8. Training of the Random Forest model on dataset 1 – Plot of the accuracy obtained on the validation data with the change in the number of estimators.

TABLE 2. Summary of Random Forest model training on the datasets

Dataset	***Validation Accuracy***
Dataset 1	0.8045
Dataset 2	0.7593
Dataset 3	0.7238
Dataset 4	0.7164
Dataset 5	0.7368

c. K-nearest Neighbors

The performance of a K-nearest neighbors classifier on the data was also considered. The hyperparameter 'k' was set as 9 after observing the results obtained on all the five datasets. The parameter 'weight' provided by the sklearn method was set to 'distance'. This ensured that the neighbors that were closer were assigned a weight greater than that of the neighbors that were further away from the test point. Thus the neighboring points had a greater influence on the final classification than the neighbors that were further away. The plot of the accuracy obtained on the validation data of dataset 1 against 'k' is shown in Fig. 9. The highest accuracy that was obtained on the validation data of each dataset after k-nearest neighbors classification is summarized in Table 3.

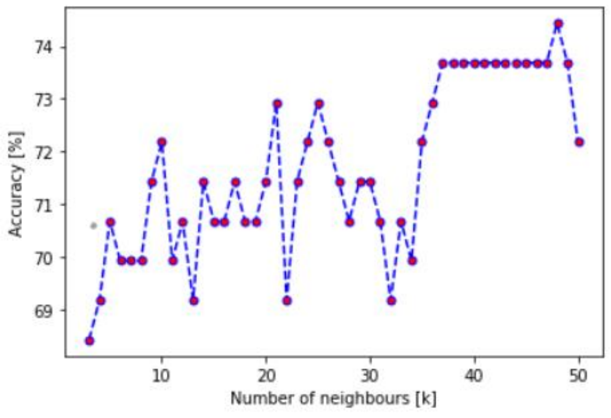


Fig. 9. Performance of the K-nearest Neighbors classifier on dataset 1 – Plot of the accuracy obtained on the validation data against the hyperparameter 'k'.

TABLE 3. Summary of K-nearest Neighbors classification on the datasets

Dataset	***Validation Accuracy***
Dataset 1	0.7443
Dataset 2	0.7368
Dataset 3	0.7164
Dataset 4	0.6194
Dataset 5	0.6917

The training results of the three classification models on the five datasets has been attached as ipynb files in the submission. The results in the file and the ones shown in the report might differ due to the randomness involved during model training.

V. Results

The results obtained on applying the trained models on the test data from each of the five datasets are given by tables 4 – 6. Each dataset contained about 134 records as test data.

TABLE 4. Performance of Neural Network Model on the testing data.

Dataset	***Accuracy***	***Precision***		***Recall***	
		No Ghost	***Ghost***	***No Ghost***	***Ghost***
Dataset 1	0.7537	0.8620	0.6710	0.6667	0.8644
Dataset 2	0.6940	0.7222	0.6612	0.7123	0.6721
Dataset 3	0.7014	0.6515	0.75	0.7166	0.6891
Dataset 4	0.6940	0.7536	0.6307	0.6842	0.7068
Dataset 5	0.6791	0.6818	0.6764	0.6716	0.686

TABLE 5. Performance of Random Forest model on the testing data.

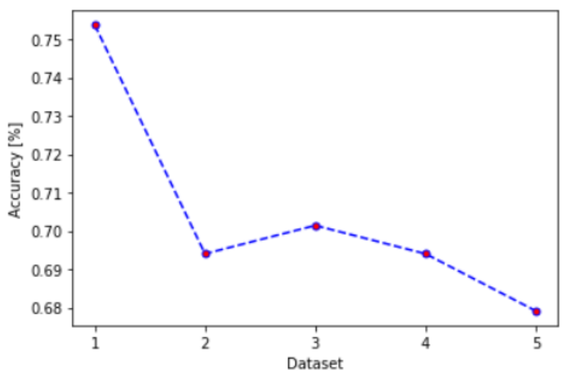
Dataset	***Accuracy***	***Precision***		***Recall***	
		No Ghost	***Ghost***	***No Ghost***	***Ghost***
Dataset 1	0.7686	0.8235	0.7121	0.7466	0.7966
Dataset 2	0.7089	0.7656	0.6571	0.6712	0.7541
Dataset 3	0.6791	0.6231	0.7384	0.7166	0.6486
Dataset 4	0.6417	0.6842	0.5862	0.6842	0.5862
Dataset 5	0.6268	0.6491	0.6103	0.5522	0.7014

TABLE 6. Performance of K-nearest Neighbor classifier on the testing data.

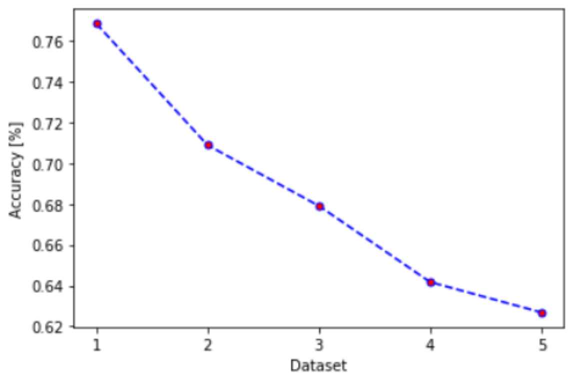
Dataset	***Accuracy***	***Precision***		***Recall***	
		No Ghost	***Ghost***	***No Ghost***	***Ghost***
Dataset 1	0.7164	0.8627	0.6265	0.5866	0.8813
Dataset 2	0.7014	0.7462	0.6567	0.6849	0.7213
Dataset 3	0.5820	0.5344	0.6184	0.5166	0.6351
Dataset 4	0.6044	0.6769	0.5362	0.5789	0.6379
Dataset 5	0.6119	0.6363	0.5949	0.5223	0.7014

VI. Discussion & Conclusion

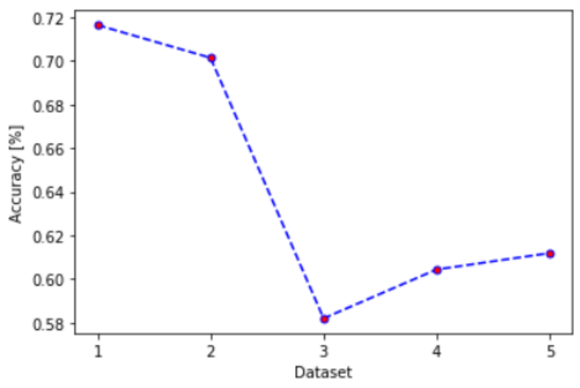
Fig. 10. shows the plot of accuracy on the test data against its corresponding dataset for each of the models trained. It could be seen that there is a decrease in accuracy as we move from dataset 1 to dataset 5. As the records from classes that are proportionally less affected by ghost prints are added, the accuracy of the model decreases. The effect could be noticed on the precision and recall values of the model too. It could be concluded that the proportion to which a fingerprint if affected by a ghost print has a direct effect on the probability of it being classified as affected by ghost print or not affected by ghost print. It is only logical that with decreased proportionality of ghost in the fingerprint the features extracted will start to resemble the features extracted from a pure print. A possible way to overcome this effect might be to try and classify each image block as either affected by ghost print or not and from those arrive at an overall output that gives the percentage of the fingerprint that is affected by ghost. This output could also help split the input fingerprint images into classes based on the severity of the ghost print. This kind of classification might need a different kind of dataset, more effort and time. Thus we propose this as an improvement to our current algorithm.



Neural Networks Classifier



Random Forest Classifier



K-Nearest Neighbors Classifier

Fig. 10. A plot of accuracy that was obtained on the test data against the corresponding dataset for each of the Machine Learning models that were trained.

References

[1] Huiden, J. E. (2019). Generating Realistic Ghost Fingerprints by Combining Real Fingerprint Images (Bachelor's thesis, University of Twente).

[2] Anil Jain. Biometrics : personal identification in networked society. Springer, New York, 2006

[3] Davide Maltoni. Handbook of fingerprint recognition. Springer, New York, 2003.

[4] M. Edwint O'Neill. The development of latent fingerprints on paper. Journal of Criminal Law and Criminology, 1937.

[5] Huang, H. C., Hsieh, C. T., Hsiao, M. N., & Yeh, C. H. (2018). A study of automatic separation and recognition for overlapped fingerprints. Applied Soft Computing, 71, 127-140.

[6] Orczyk, T. and Wieclaw, L., 2011, October. Fingerprint ridges frequency. In *2011 Third World Congress on Nature and Biologically Inspired Computing* (pp. 558-561). IEEE.

[7] X. Zhan, Z. Sun, Y. Yin and Y. Chu, Fingerprint Ridge Distance Estimation: Algorithms and the Performance, ICB 2006, pp. 294-301, 2006.

[8] Chen, F., Feng, J., Jain, A.K., Zhou, J. and Zhang, J., 2011. Separating overlapped fingerprints. *IEEE Transactions on Information Forensics and Security*, *6*(2), pp.346-359.

[9] Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.