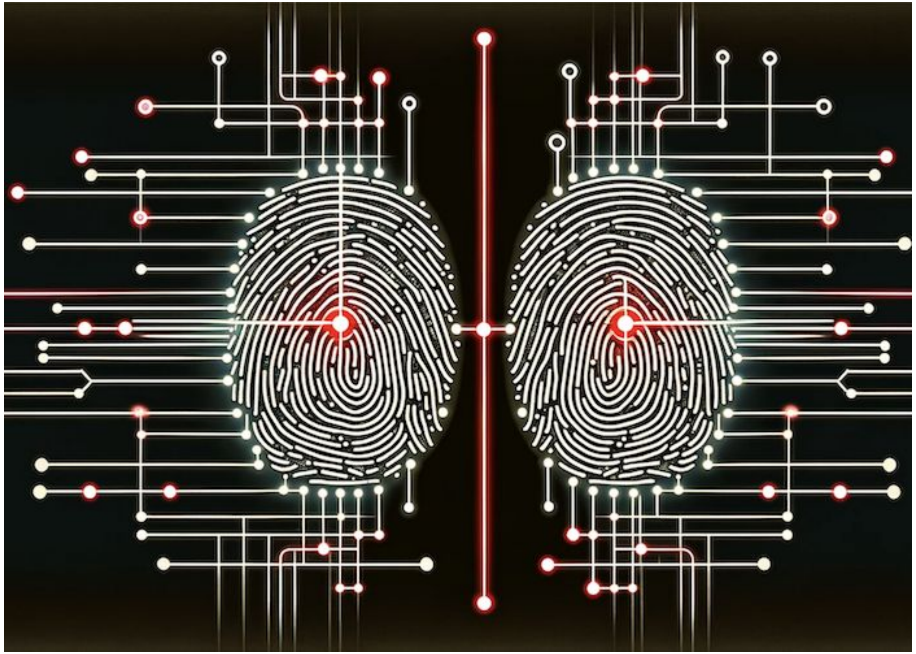
FINGERPRINT IDENTIFICATION

MACHINE LEARNING

Epoka University
Department of Computer Engineering
Computer Engineering

CONTENT PAGE



INTRODUCTION

- One of the most important components of modern security and identity systems is fingerprint recognition.
- Our main goal is to use various machine learning methods to achieve reliable fingerprint identification.
- We will investigate many algorithms along the process, including : *Convolutional Neural Networks (CNN), Multi-Layer Perceptrons (MLP), k-Nearest Neighbors (k-NN), and Support Vector Machines (SVM).*

- A key point in our process will also be image **PRE - PROCESSING.**

METHODOLOGY

❖ **Three main steps:**

1. Image preprocessing, involving normalization techniques and feature extraction.

2. Training models with distinct structures or parameters (e.g., MLP with varying hidden layer nodes).

3. Evaluating models and computing accuracy metrics to identify the best-performing model under different circumstances.

DATASET

- Images of 8 fingerprint captures of 16 different fingers. In total there are 128 images.
- Their dimensions are 248px width and 338px height.
- Naming convention → first we have the “label” (ex. “8”) of the image, signifying which finger it is, followed by an “_” and then the specific fingerprint capture number, from 1 to 8.

STATE OF ART

Methodology	Author	Accuracy
SVM	Xuan Xu	90.24%
SVM	S. Adebayo Daramola	80%
CNN	Bhavesh Pandya	98.21%
SVM	Youssef Elmir	68.4%
MLP	Terje Kristensen	88.8%

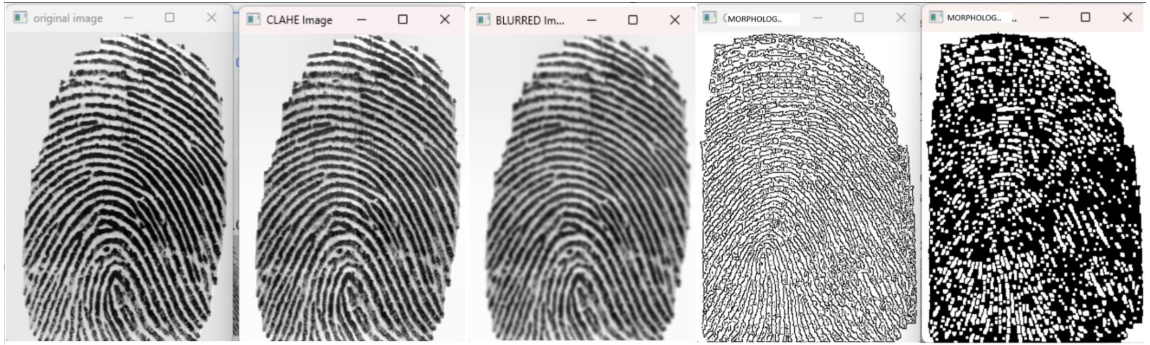
- Existing papers focus on CNNs and SVMs, lacking detailed implementation steps.
- Some emphasize feature extraction, crucial for overall performance, while others delve into complex math, challenging for students.
- Our approach: Draw inspiration from existing works and create an all in one project.

DATA PREPROCESSING

Fingerprint recognition heavily relies on the quality of input images, making preprocessing crucial for optimal algorithm performance. We detail our exploration of preprocessing steps to enhance image quality and improve our identification program, iterating through various pipelines until successful.

FIRST PREPROCESSING PIPELINE:

1. Resizing Images
2. Applying CLAHE
3. Noise Reduction:
4. Edge Enhancement:
5. Binary Thresholding:
6. Morphological Operations:



RESULTS OF FIRST PIPELINE

After testing, we found that using the CLAHE image improved the contrast, while the image after morphological analysis became unreadable. Despite trying the SVM algorithm with this pipeline, the results were unsatisfactory, with processed data accuracy almost identical to raw data.

Classification Report:

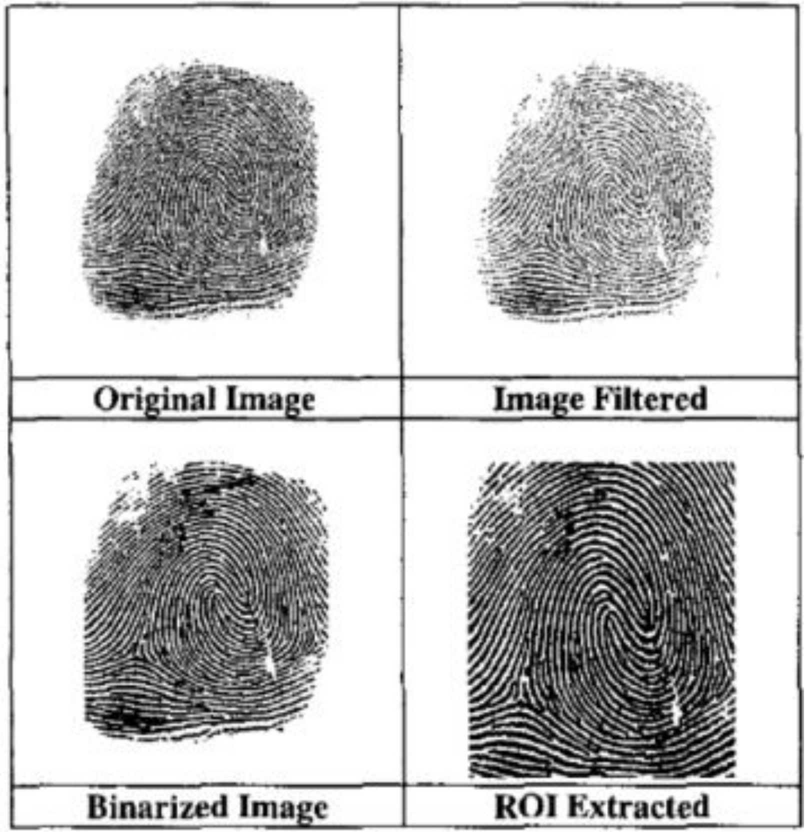
	precision	recall	f1-score	support
1	0.00	0.00	0.00	1
2	0.50	0.50	0.50	2
3	0.00	0.00	0.00	0
4	1.00	0.67	0.80	3
5	1.00	1.00	1.00	1
6	0.00	0.00	0.00	2
7	0.33	0.50	0.40	2
8	0.00	0.00	0.00	2
9	0.50	1.00	0.67	1
10	0.00	0.00	0.00	1
11	0.25	0.50	0.33	2
12	1.00	0.50	0.67	2
13	0.00	0.00	0.00	3
14	0.00	0.00	0.00	1
15	0.50	0.50	0.50	2
16	1.00	1.00	1.00	1
accuracy			0.38	26
macro avg	0.38	0.39	0.37	26
weighted avg	0.41	0.38	0.38	26

Classification Report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	1
2	0.50	0.50	0.50	2
3	0.00	0.00	0.00	0
4	1.00	0.67	0.80	3
5	1.00	1.00	1.00	1
6	0.00	0.00	0.00	2
7	0.00	0.00	0.00	2
8	1.00	0.50	0.67	2
9	0.50	1.00	0.67	1
10	0.00	0.00	0.00	1
11	0.25	0.50	0.33	2
12	1.00	0.50	0.67	2
13	0.00	0.00	0.00	3
14	0.00	0.00	0.00	1
15	0.50	0.50	0.50	2
16	1.00	1.00	1.00	1
accuracy			0.38	26
macro avg	0.42	0.39	0.38	26
weighted avg	0.46	0.38	0.40	26

SECOND PIPELINE

1. CLAHE application
2. Gaussian blur (3x3 size)
3. Binarize through Otsu's method
4. Thinning to enhance ridges
5. Region of Interest ROI



RESULTS OF SECOND PIPELINE

Unfortunately, the last approach yielded poor results, with the SVM accuracy of the ROI version at 0.19. Surprisingly, binarized images performed better, and raw images without preprocessing performed even better.

SVM RESULTS WITH THINNED SQUARED ROI IMAGES

Classification Report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	1
2	1.00	0.50	0.67	2
3	0.00	0.00	0.00	0
4	0.00	0.00	0.00	3
5	0.00	0.00	0.00	1
6	0.00	0.00	0.00	2
7	0.00	0.00	0.00	2
8	0.00	0.00	0.00	2
9	0.00	0.00	0.00	1
10	0.00	0.00	0.00	1
11	0.50	0.50	0.50	2
12	0.20	0.50	0.29	2
13	0.00	0.00	0.00	3
14	1.00	1.00	1.00	1
15	0.00	0.00	0.00	2
16	1.00	1.00	1.00	1
accuracy			0.19	26
macro avg	0.23	0.22	0.22	26
weighted avg	0.21	0.19	0.19	26

SVM RESULTS WITH BINARIZED IMAGES

Classification Report:

	precision	recall	f1-score	support
1	0.00	0.00	0.00	1
2	0.50	0.50	0.50	2
3	0.00	0.00	0.00	0
4	0.00	0.00	0.00	3
5	0.50	1.00	0.67	1
6	0.00	0.00	0.00	2
7	0.50	1.00	0.67	2
8	1.00	0.50	0.67	2
9	1.00	1.00	1.00	1
10	0.00	0.00	0.00	1
11	0.50	0.50	0.50	2
12	1.00	0.50	0.67	2
13	0.00	0.00	0.00	3
14	0.20	1.00	0.33	1
15	0.50	0.50	0.50	2
16	1.00	1.00	1.00	1
accuracy			0.42	26
macro avg	0.42	0.47	0.41	26
weighted avg	0.41	0.42	0.38	26

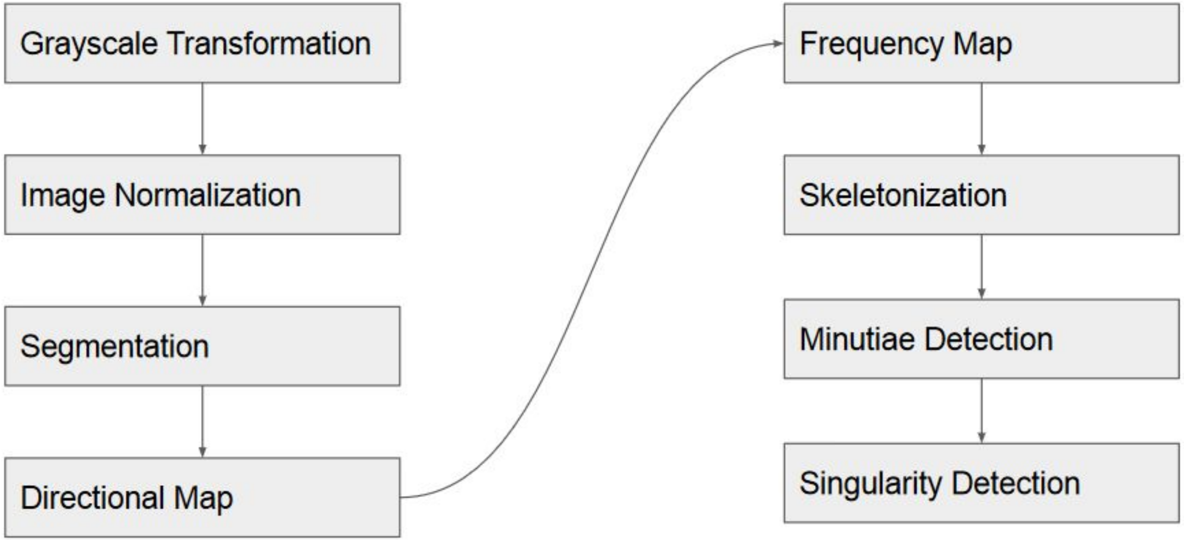
WITHOUT ANY PREPROCESSING ://///////////////////////// (better)

Classification Report:

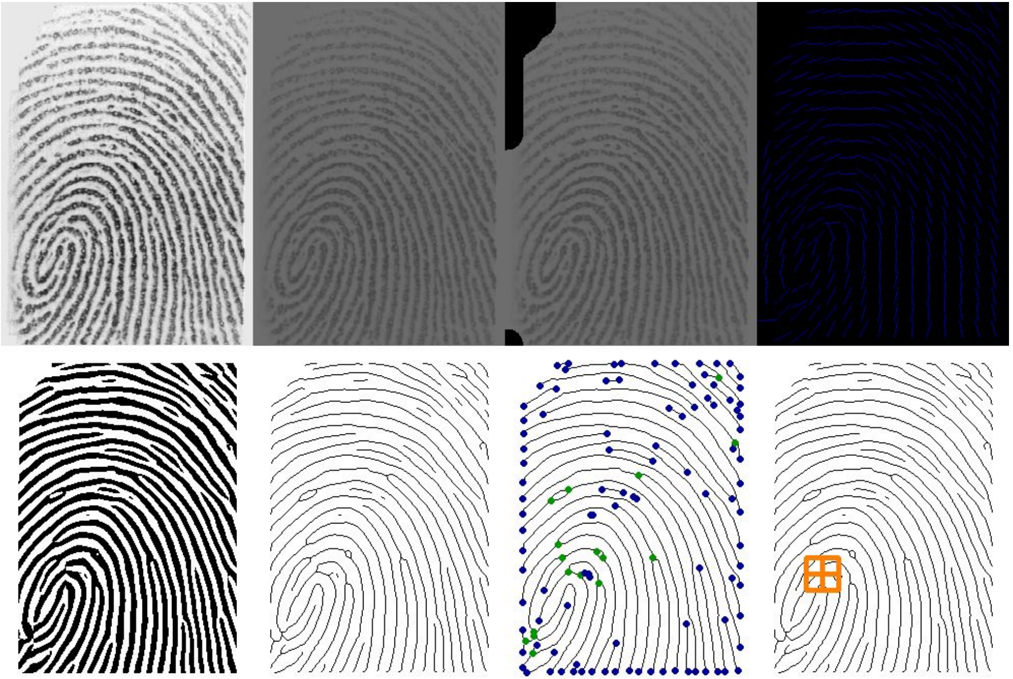
	precision	recall	f1-score	support
1	0.00	0.00	0.00	1
2	0.50	0.50	0.50	2
4	1.00	1.00	1.00	3
5	0.50	1.00	0.67	1
6	1.00	1.00	1.00	2
7	0.33	0.50	0.40	2
8	1.00	0.50	0.67	2
9	1.00	1.00	1.00	1
10	0.00	0.00	0.00	1
11	1.00	0.50	0.67	2
12	1.00	1.00	1.00	2
13	1.00	0.33	0.50	3
14	0.50	1.00	0.67	1
15	0.50	1.00	0.67	2
16	1.00	1.00	1.00	1
accuracy			0.69	26
macro avg	0.69	0.69	0.65	26
weighted avg	0.76	0.69	0.68	26

THIRD PIPELINE

After reviewing successful works by other engineers, we opted to adopt the feature extraction approach outlined by Manuel Cuevas [4]. Minutiae extraction plays a vital role in fingerprint recognition. Below are the stages of our preprocessing pipeline:



RESULTS OF THE THIRD PIPELINE



- Generated images at each step to show changes
- Merged images to visualize complete progress
- Processed images maintained clarity and identification information
- Minutiae section highlighted endings and bifurcations
- Attempted to save minutiae coordinates for feature vector creation. Wasn't able to put the saved data to use
- Opted to use minutiae images directly for training and testing
- Saved minutiae images separately for easy access across algorithms

SVM

The SVM model, with parameters including kernel, achieved a peak accuracy of 0.69 using a linear kernel on original images.

This suggests that the chosen preprocessing was not optimal for this algorithm, as the second- highest accuracy was also attained with original images. Minutiae images with third preprocessing and binarized images with second preprocessing came after.

The polynomial kernel yielded an accuracy of 0.0, rendering it invalid for this case

	Accuracy	Precision	Recall	F1 Score
First preprocessing, linear kernel	0.38	0.39	0.37	0.38
First preprocessing, poly kernel	0.0	0.0	0.0	0.0
Second preprocessing, ROI images	0.19	0.22	0.19	0.19
Second preprocessing, binarized images	0.42	0.47	0.41	0.38
Original Images, linear kernel	0.69	0.76	0.68	0.65
Third preprocessing, Minutiae images	0.44	0.54	0.44	0.45
Third preprocessing, Original images	0.55	0.77	0.55	0.59

MLP

When implementing a Multilayered Perceptron Neural Network, we adjust parameters like the number of nodes in the hidden layer and the activation function (ReLu in our case).

We trained the MLP model on both Minutiae and original images with varying hidden layer nodes. Due to limited computational power, we ran the algorithm with 100 epochs.

After correcting a mistake in data splitting, the 128-node architecture achieved 0.44 accuracy with Minutiae data. For original data, accuracy was 0.39 with 128 nodes and **0.5 with 256 nodes.**

```
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step - accuracy: 1.0000 - loss: 0.0399 - val_accuracy: 0.3333 - val_loss: 2.1345
Epoch 96/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 57ms/step - accuracy: 1.0000 - loss: 0.0369 - val_accuracy: 0.3889 - val_loss: 2.1425
Epoch 97/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step - accuracy: 1.0000 - loss: 0.0347 - val_accuracy: 0.2222 - val_loss: 2.1921
Epoch 98/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 60ms/step - accuracy: 1.0000 - loss: 0.0371 - val_accuracy: 0.2222 - val_loss: 2.2030
Epoch 99/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 55ms/step - accuracy: 1.0000 - loss: 0.0370 - val_accuracy: 0.2778 - val_loss: 2.1091
Epoch 100/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 0s 58ms/step - accuracy: 1.0000 - loss: 0.0342 - val_accuracy: 0.4444 - val_loss: 2.0853
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.4444 - loss: 2.0853
Test accuracy: 0.4444444477558136
```

	128 nodes in hidden layer	256 nodes	512 nodes
Original Images	0.11	0.28	0.39
Minutiae Images	0.22	0.33	0.28

KNN

	3-neighbors	5-neighbors	7-neighbors	9-neighbors
Minutiae Images	0.28	0.33	0.33	0.38
Original Images	0.17	0.17	0.11	0.05

In this algorithm, the only adjustable parameter is the number of neighbors (k). Despite low accuracy, processed images outperform original ones, with the highest performance observed at k=9 neighbors.

RANDOM FOREST

Experimented with varying the "number of estimators" and found the best result (though still unsatisfactory) with 50 and 200 as the numbers. Interestingly, the highest accuracy for this algorithm was observed again when using the original images.

	Nr. of Estimators	Accuracy	Precision	Recall	F1 Score
Minutiae Images	**100**	0.17	0.34	0.17	0.18
	200	0.22	0.31	0.22	0.21
	250	0.28	0.48	0.28	0.29
Original Images	**100**	0.44	0.32	0.44	0.34
	200	0.5	0.46	0.5	0.43
	250	0.5	0.46	0.5	0.43

HYBRID

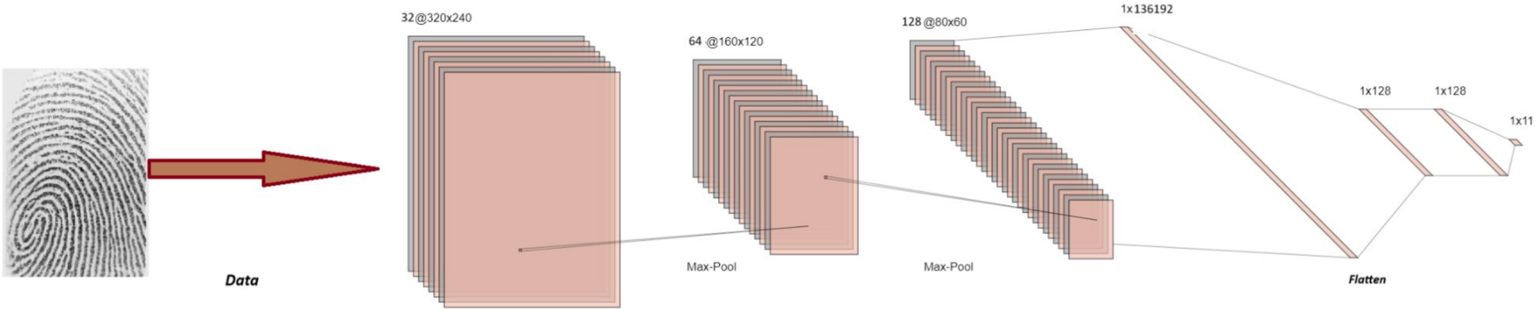
We attempted to create a hybrid model by combining the results of our KNN and Random Forest models through majority voting. We selected the versions of our base models with the highest accuracy: KNN with k=9 using minutiae images and Random Forest with 200 estimators using original images. However, this hybrid model performed worse than both previous algorithms, resulting in lower accuracy.

```
Hybrid Accuracy: 0.2222222222222222
```

CNN-1

FIRST PROPOSED ARCHITECTURE:

- 3 convolutional layers, each of them with 32, 64 & 128 nodes respectively
- Two dense hidden layers with 128 nodes each
- Windowing size is 3x3
- ReLu activation function, on last classification layer → “softmax”
- Adam optimizer.



Using the Minutiae Images

Accuracy: 0.89 (best)

```
Epoch 98/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 393ms/step - accuracy: 0.9890 - loss: 0.0781 - val_accuracy: 0.8889 - val_loss:
0.9159
Epoch 99/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 414ms/step - accuracy: 0.9779 - loss: 0.1122 - val_accuracy: 0.9444 - val_loss:
0.5657
Epoch 100/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 390ms/step - accuracy: 0.9597 - loss: 0.1980 - val_accuracy: 0.8889 - val_loss:
0.3118
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 81ms/step - accuracy: 0.8889 - loss: 0.3118
Test Loss: 0.311845988035202
Test Accuracy: 0.8888888955116272
```

Using the Original Images

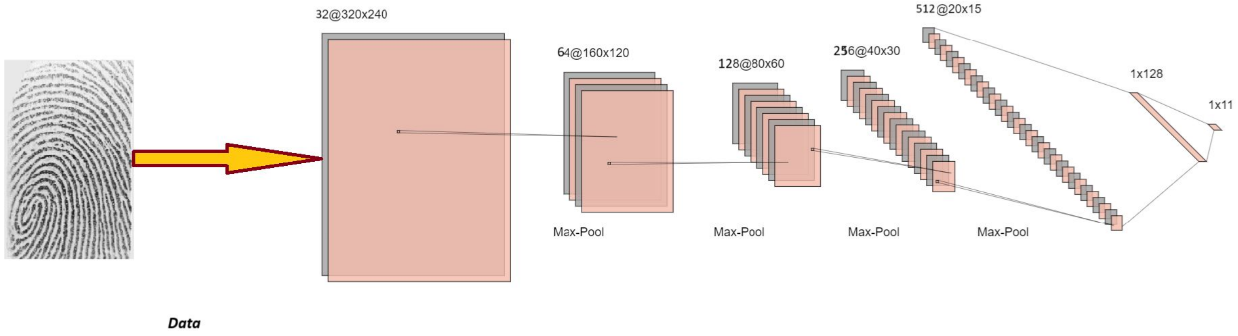
Accuracy: 0.78

```
0.8349
Epoch 97/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 3s 1s/step - accuracy: 1.0000 - loss: 0.0206 - val_accuracy: 0.7778 - val_loss: 0.6637
Epoch 98/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 3s 958ms/step - accuracy: 1.0000 - loss: 0.0074 - val_accuracy: 0.7778 - val_loss:
0.5601
Epoch 99/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 4s 1s/step - accuracy: 0.9890 - loss: 0.0500 - val_accuracy: 0.8889 - val_loss: 0.4218
Epoch 100/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 3s 1s/step - accuracy: 1.0000 - loss: 0.0105 - val_accuracy: 0.7778 - val_loss: 0.4073
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 172ms/step - accuracy: 0.7778 - loss: 0.4073
Test Loss: 0.4073316752910614
Test Accuracy: 0.7777777910232544
```

CNN-2

SECOND PROPOSED ARCHITECTURE:

- 5 convolutional layers, each of them with 32, 64, 128, 256 & 512 nodes respectively
- Two dense hidden layers with 128 nodes each
- Windowing size is 3x3
- ReLu activation function, on last classification layer → "softmax"
- Adam optimizer.



Using the Minutiae Images

Accuracy: 0.83

```
Epoch 98/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 409ms/step - accuracy: 1.0000 - loss: 0.0248 - val_accuracy: 0.7778 - val_loss:
1.4292
Epoch 99/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 420ms/step - accuracy: 0.9701 - loss: 0.0737 - val_accuracy: 0.8333 - val_loss:
1.3854
Epoch 100/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 410ms/step - accuracy: 0.9811 - loss: 0.0360 - val_accuracy: 0.8333 - val_loss:
1.3149
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 108ms/step - accuracy: 0.8333 - loss: 1.3149
Test Loss: 1.3149492740631104
Test Accuracy: 0.8333333134651184
```

Using the Original Images

Accuracy: 0.72

```
Epoch 96/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 2s 437ms/step - accuracy: 0.9701 - loss: 0.1690 - val_accuracy: 0.7222 - val_loss: 1.4628
Epoch 97/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 426ms/step - accuracy: 0.9181 - loss: 0.1860 - val_accuracy: 0.7222 - val_loss: 1.4586
Epoch 98/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 433ms/step - accuracy: 0.9142 - loss: 0.3722 - val_accuracy: 0.7222 - val_loss: 1.3664
Epoch 99/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 1s 421ms/step - accuracy: 0.9324 - loss: 0.1591 - val_accuracy: 0.7222 - val_loss: 1.3852
Epoch 100/100
3/3 ━━━━━━━━━━━━━━━━━━━━ 2s 439ms/step - accuracy: 0.9512 - loss: 0.1378 - val_accuracy: 0.7222 - val_loss: 1.5296
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 130ms/step - accuracy: 0.7222 - loss: 1.5296
Test Loss: 1.5295895338058472
Test Accuracy: 0.7222222089767456
```

FULL TABLE OF RESULTS

	Pre-processing	Specifications	Accuracy
SVM	None	Linear kernel	0.69
	Third (minutiae)	Linear kernel	0.44
	First	Linear kernel	0.38
MLP	Third (minutiae)	128 nodes in hidden layer	0.22
	Third (minutiae	256 nodes in hidden layer	0.5
	None	512 nodes in hidden layer	0.39
KNN	Third (minutiae)	9 - neighbors	0.38
	Third (minutiae)	5 - neighbors	0.33
	None	3 - neighbors	0.17
Random Forest	None	200 nr of estimators	0.5
	Third (minutiae)	200 nr of estimators	0.22
	None	100 nr of estimators	0.44
Hybrid		KNN + Random Forest	0.22
CNN	Third (minutiae)	3 convolutional layers	**0.89**
	Third (minutiae)	5 convolutional layers	0.83
	None	3 convolutional layers	0.77
	None	5 convolutional layers	0.72

CONCLUSION

The top-performing model was the Convolutional Neural Network (CNN) with 3 hidden convolutional layers and 2 dense hidden layers, with ReLu activation function and Adam optimizer, achieving an 89% accuracy. The effectiveness of this model was facilitated by the initial preprocessing pipeline applied to the images. Notably, original images without preprocessing yielded a maximum accuracy of 77%. To our surprise, the Random Forest model achieved a 50% accuracy, defying expectations given the lack of prior literature suggesting its efficacy. The hybrid model also wasn’t expected to perform that poorly but it requires further investigation.