**Capstone Report – u3254555 Dihela Gamaethige**

**Introduction**

In today's world, biometrics—the study of identifying people based on their distinct physiological or behavioural traits—plays an increasingly important role in everything from border control systems to smartphone security. Even though biometrics research has advanced significantly in recent years, large-scale and diverse datasets are still essential for training and testing biometric algorithms. Under this light, the Sokoto Coventry Fingerprint Dataset, or SOCOFing, stands out as a novel tool designed for scholarly study and intended to further the biometrics community.

A significant advancement in the field of fingerprint identification, SOCOFing provides 6,000 carefully curated fingerprint photos from 600 African participants. These participants were picked to reflect the rich mosaic of biometric variety found in Africa, a feature that is frequently lacking in datasets currently in use.

One of SOCOFing's unique features is its extensive annotation system, which provides researchers with a wealth of useful data for in-depth investigation and algorithm improvement. The collection includes labels for crucial details like gender, hand (left or right), and finger name, which helps to clarify how fingerprint variation varies along these dimensions.

The continuous worldwide attempts to provide more inclusive, varied, and realistic biometric datasets are demonstrated by SOCOFing. By concentrating on the African population, it not only helps to portray human variety more fairly but also encourages biometric technology innovation and resilience. In order to improve the area of fingerprint identification and tackle the particular difficulties associated with the diversity of African biometrics, biometrics researchers and practitioners are well-positioned to take advantage of the SOCOFing dataset. This will ultimately contribute to a more secure and inclusive global community.

**Stage 1 EDA**

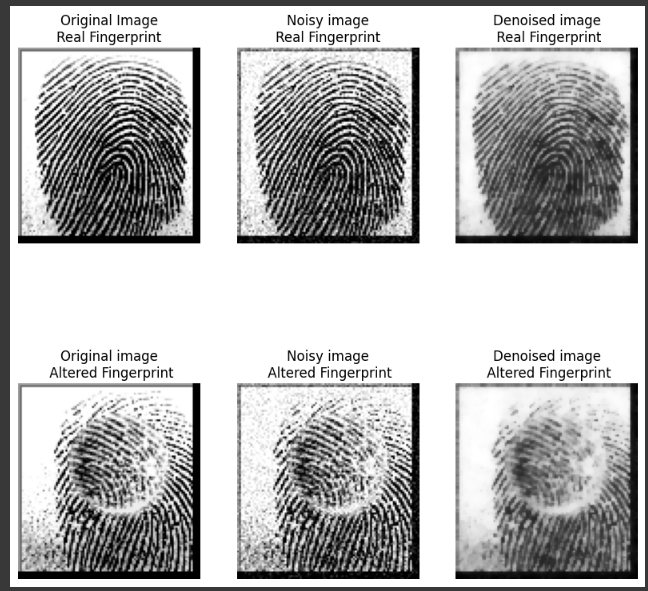
1. How can two images be displayed?



1. How can noise be introduced to manipulate an image?

A screenshot of a computer program

Description automatically generated



1. How can blur be used to manipulate an image?

A computer screen shot of text

Description automatically generated

Several different types of fingerprints

Description automatically generated

1. How can we detect corners on an image?

A screenshot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

1. How can we detect corners on a grayscale image?

A screenshot of a computer program

Description automatically generated

**A close-up of a fingerprint

Description automatically generated**

**Stage 2 PDA**

from keras.models import load\_model # TensorFlow is required for Keras to work  
from PIL import Image, ImageOps # Install pillow instead of PIL  
import numpy as np  
  
# Disable scientific notation for clarity  
np.set\_printoptions(suppress=True)  
  
# Load the model  
model = load\_model("keras\_Model.h5", compile=False)  
  
# Load the labels  
class\_names = open("labels.txt", "r").readlines()  
  
# Create the array of the right shape to feed into the keras model  
# The 'length' or number of images you can put into the array is  
# determined by the first position in the shape tuple, in this case 1  
data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)  
  
# Replace this with the path to your image  
image = Image.open("Altered-Easy.BMP").convert("RGB")  
  
# resizing the image to be at least 224x224 and then cropping from the center  
size = (224, 224)  
image = ImageOps.fit(image, size, Image.Resampling.LANCZOS)  
  
# turn the image into a numpy array  
image\_array = np.asarray(image)  
  
# Normalize the image  
normalized\_image\_array = (image\_array.astype(np.float32) / 127.5) - 1  
  
# Load the image into the array  
data[0] = normalized\_image\_array  
  
# Predicts the model  
prediction = model.predict(data)  
index = np.argmax(prediction)  
class\_name = class\_names[index]  
confidence\_score = prediction[0][index]  
  
# Print prediction and confidence score  
print("Class:", class\_name[2:], end="")  
print("Confidence Score:", confidence\_score)