# COMP6211 Biometrics Coursework

Name: Shenghao Li   ID: 33091374

***Abstract*—Biometrics are biological measurements or physical characteristics that can be used to identify individuals. For example, fingerprint mapping, facial recognition, and retina scans are all forms of biometric technology. In this report, a system is designed to recognize people according to their body shape. Images are provided by the Large Southampton Gait Database [1]. A body pose tracking model is derived using the training set images based on a Machine Learning model called MediaPipe Pose. The first rank Correct classification rate reached 72.7% with an EER threshold of 0.1048.**

***Index Terms*—Body shape, MediaPipe Pose, Pose recognition, Machine Learning, EER, Correction Classification Rate, Pose Estimation**

## I. Introduction

Biometrics are body measurements and calculations related to human characteristics. Biometric authentication is used in computer science to identify and access control. Biometric identifiers are often categorized as physiological characteristics, including fingerprint, hand geometry, face recognition, etc. Considering the limited quantity of training images, the initial approach is to fit a virtual skeleton on a human body, also known as pose estimation. We first extract the features for each subject by using MediaPipe Pose. Then we compare the characteristics of each test subject with all the training sets in the database, making identification based on the minimum Euclidean Distance. Then, we adjust the weight value to achieve better Correction Classification Rate performance. Finally, we display the histogram of distances between the subjects and justify the recognition algorithm regarding its CCR and EER performance.

Figure. 1: An example of Front and Side images in the training set

## II. Method

This section tried different virtual skeleton fitting models and compared their performances. We first tried MobileNet V2 architecture by using movenet/singlepose/thunder model from TensorFlow Hub. It is a convolutional neural network model that runs on RGB images and predicts the human joint locations of a single person. This model improves the state-of-the-art performance of mobile models on multiple tasks and benchmarks and across a spectrum of different model sizes [2]. The result is shown below.

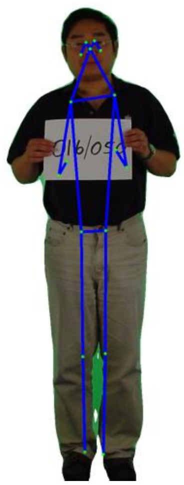

Figure. 2: Key Points Extraction by MobileNet V2

As we can see, it has a bad performance for elbow joints and hands. Thus, it is not suitable for this project.
Then, we tried the DeepPose model, which makes the human pose estimation vis Deep Neural Networks and is formulated as a DNN-based regression problem toward body joints [3]. We used 19 key points for extraction. However, it doesn't work well with the eyes, nose, and left leg, and we didn't have key points for feet, which is essential for the classification. The result is shown below.


Figure. 3: Key Points Extraction by DeepPose

### A. *Feature Extraction by MediaPipe Pose*

MediaPipe Pose is a Machine Learning solution for high-fidelity body pose tracking, inferring 33 3D landmarks, and background segmentation mask on the whole body from RGB video frames utilizing BlazePose research that also powers the ML Kit Pose Detection API. Inspired by Leonardo's Vitruvian man, the BlazePose model predict the midpoint of a person's hips, the radius of a circle circumscribing the whole person, and the incline angle of the line connecting the shoulder and hip midpoints [4].
The landmark model in MediaPipe Pose predicts the location of 33 pose landmarks. It can also predict a full-body segmentation mask represented as a two-class segmentation. The key points figure is shown below.

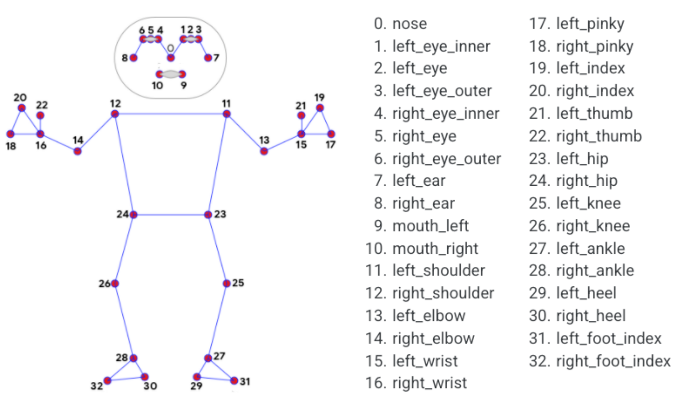


Figure. 4: 33 pose landmarks

By importing the mediapipe library, we can create a pose subject with detection confidence and tracking confidence of 0.9 from the mediapipe pose class. Then we make the detection by invoking the process method of pose and populating the training set. This way, we can extract the landmarks of the image. We can derive the virtual skeleton by rendering the landmarks on the image, as shown below.


Figure. 5: Front-on and Side-on Extraction by MediaPipe

Then, by using the x and y coordinates of the extracted key points, we selected the features and adjusted their weight value. As for front views, we used the length of eyes, the distances between the nose and all the symmetric key points on the body, length of shoulder, length of the hip, length of the right leg, and length of the right upper arm. As for side-on views, we used the length of the left arm, length of the left leg, length of the left foot, and distances between the nose to the left side of symmetric key points on the body. Through trial and error, we found that by adding more considerable weights value to small features, the performance of this algorithm can be significantly improved. We can derive a multidimensional feature vector for each subject by combining all the features together.
Then, we created a Database function to store all the features extracted from each subject in the training set into a list type variable.

### B. *Identification*

We first match the test subject with its related subject in the training set manually and store their index in a list; then, by comparing each test subject with the database features, we selected the shortest Euclidean distance between the test subject feature vector and the feature vectors in the database, which leads to the recognition result. Walking through all the test subjects, we can get the Correct Classification Rates for the first rank as 72.7%, which means we got 8 correct identifications out of 10. The performance is pretty good, considering we just have two images and one sample for a subject.

```
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
8
PS D:\File\BodyshapeRecognition>
```

Figure. 6: Identification result is shown in the Terminal Interface of VSCode

## III. RESULT

The Correct Collection Rate reached 72.7% for the first rank Euclidean distance. Then, we tried to show the histograms of distances between each test subject and all the subjects in the training set. We got some graphs as shown below.

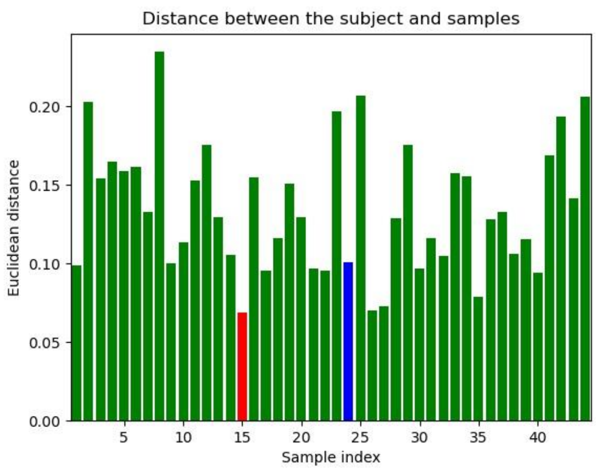


Figure. 7: First test subject example

Here, we use the blue column to denote the right identification and the red column for the estimated identification if the recognition result is wrong. So, we didn't get the correct classification for the first test subject.

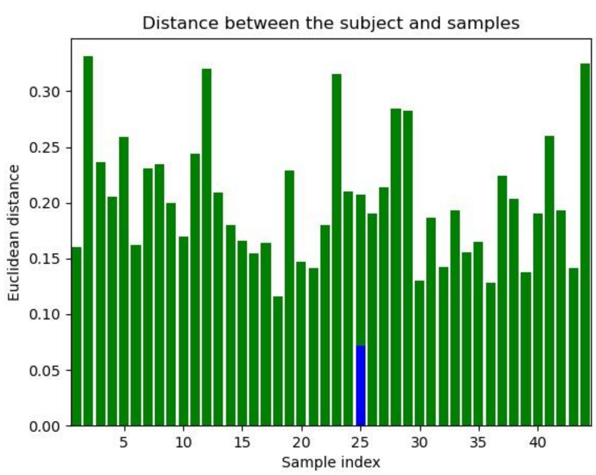


Figure. 8: Second test subject example

As we can see in the graph, the identification result is correct for the second test subject.

To provide the Equal Error Rate for subject verification, we first need to understand some terminologies.

Equal error rate (EER) is a biometric security system algorithm used to predetermine the threshold values for its acceptance rate and its rejection rate. When the rates are equal, the expected value is referred to as the equal error rate.

False rejection rate (FRR) is the measure of the likelihood that the biometric security system will incorrectly reject an access attempt by an authorized user.

False acceptance rate (FAR) is a unit used to measure the average number of false acceptances within a biometric security system. It measures and evaluates the efficiency and accuracy of a biometric security system by determining the rate at which unauthorized or illegitimate users are verified on a particular system.

So, we need to make FRR equal to FAR, which can be calculated using the equations below.

$$FAR = \frac{NFA}{NIRA} \times 100\%$$
$$FRR = \frac{NFR}{NGRA} \times 100\%$$

Here, NIRA denotes matching times between different classes, NFA denotes the number of false acceptance times, NGRA denotes matching times inside the same class, and NFR means the number of false rejection times.

We need to make some adjustments to our algorithm. Instead of using first rank Euclidean distance, we should use a concrete Euclidean distance value as the threshold. If the tested distance is bigger than this value, we assume they don't match with each other; else, they are from the same class. The NIRA value under our scenario is 11*43, while the NGRA is 11 (considering all the test subjects)

Then, we can get the following result.

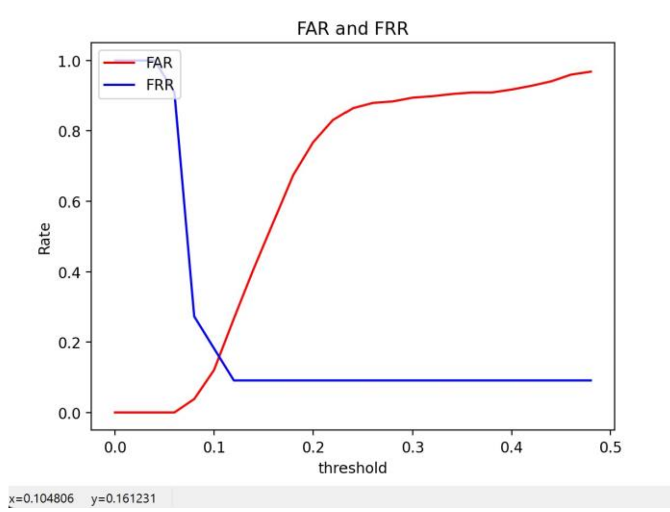


Figure. 9: FAR and FRR

As shown in the graph, FAR increases from 0 to 1 as the range of threshold grows. This is because we are likely to falsely accept a subject when the threshold is high. In contrast, FRR decreases from 1 to 0 because it's hard to make false rejections when the threshold is high.

The intersection point of the two curves is (0.104806, 0.161231). This means that when the threshold is 0.104806, we can provide Equal Error Rates for the subject verification system.

## IV. DISCUSSION

Since we just have two images for a single subject in the training set, I extract features from both the front and side views to fully use all the information. This means that we have only one feature vector for each subject in the training set, which increases the complexity of the recognition. By adjusting the selected features and their weights, we can reach a very high-level recognition rate but face over-fitting simultaneously, which means if a new subject is added to the testing set, the recognition rate is hard to sustain at that level. Also, since we just have one sample for each class, KNN is converted to find the minimum Euclidean distance between a test feature and features in the database.

## V. CONCLUSION

Through this coursework, I designed a human recognition system based on body shape, which is easier to extract features than fingerprint or face recognition. It is an easy and low-cost way to make the classification. With only two images for each subject, this system reached a CCR of 72.7% for the first rank Euclidean distance.

## REFERENCES

[1] Shutler, J.D., Grant, M.G., Nixon, M.S., Carter, J.N. (2004). On a Large Sequence-Based Human Gait Database. In: Lotfi, A., Garibaldi, J.M. (eds) Applications and Science in Soft Computing. Advances in Soft Computing, vol 24. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-45240-9_46

[2] Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[3] Toshev, Alexander, and Christian Szegedy. "Deeppose: Human pose estimation via deep neural networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[4] Bazarevsky, V. and Grishchenko, I., 2022. On-device, Real-time Body Pose Tracking with MediaPipe BlazePose. [online] Google AI Blog. Available at:<https://ai.googleblog.com/2020/08/on-device-real-time-body-pose-tracking.html> [Accessed 13 May 2022].