Preventing Turtle Neck Syndrome using Machine Learning

Jongwon Won[1] and Dongwoo Lee[2]

Sungkyunkwan University, Media Communication
Sungkyunkwan University, Mechanical Engineering

Abstract. Turtle Neck Syndrome is a major disease that affects modern people in a sedentary society like ours. Turtle neck not only looks bad, but it can also cause psychological distress as well as physical discomfort. The best way to avoid this is to make it a habit to maintain proper posture on a regular basis. However, when you concentrate on your work, your posture changes unconsciously. As a result, we offer a solution that monitors the user's posture in real time and alerts the user when he or she is wearing a turtle neck to correct the posture. Pose estimation with OpenPose and user posture determination with an artificial neural network are used for this purpose.

Keywords: Turtle Neck Syndrome · Deep Learning · AWS

1 Introduction

People in the twenty-first century are subjected to major postural imbalances. There are a variety of reasons for this, including muscular weakness that corrects body posture owing to a lack of exercise or carrying a large bag, but the most common explanation is sitting for the majority of the day. As a result, humans prefer to sit in a comfortable position, even though that position is occasionally unhealthy. A faulty posture with the chin tucked or the neck projecting forward, for example, is a common occurrence. Long periods of time spent in an imbalanced posture are harmful to the human body. It creates symptoms like scoliosis in moderate cases, but in severe situations, it can lead to diseases like disc or temporomandibular joint issues. Of course, people with serious joint problems should seek professional care, but by altering our daily habits, we can help you maintain proper posture. To put it another way, we attempted to devise a method for using picture data to monitor and alter human posture in real time in order to address the issue of 'unbalanced posture.' We wanted to create a system that recognizes the disadvantageous position known as "Turtle Neck Syndrome" and prevents humans from adopting it. As a result, we developed an app that detects whether a human location in image data is unsuitable and alerts the user. There are two types of primary technologies that are necessary for this.
The first technique is Pose Estimation. This is not to be confused with postural diagnosis. Pose Estimation refers to the behavior of the machine learning model

in this study that 'discovers imbalanced postures.' This method derives human posture from picture and video data. Key aspects of the face, such as the joints and eyes, can be extracted and represented as coordinates, forming the core of the human body. The second technology is Deep Learning. Deep learning is a subset of machine learning that uses artificial neural networks as its foundation. In addition to the input and output layers, a neural network has several hidden layers in the middle. Each node is fully connected to the others and has its own value. During the training process, the neural network modifies the value of each node.
We developed an application for this purpose with the goal of using these two technologies to help users correct their posture by preventing them from taking a bad posture such as a turtle neck. The figure below is an example of the mobile application of this project. The mobile application and desktop application communicate with the server in real time to determine the posture and exchange information, and when the result of the posture determination is delivered, it informs the user.

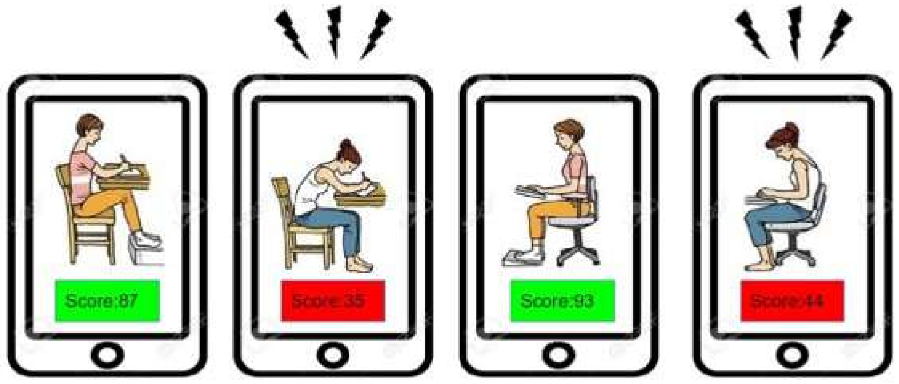


Fig. 1. This is a mock-up of an app that can tell when a user is in good posture and when they aren't. The shooting angle is set to the front in the actual program, and instead of displaying the user's posture score on the screen, a notice message with vibration is presented only when the user takes the inappropriate posture. The non-score is used to ensure that the application does not distract the user from their task.

2 Design for the service

2.1 Overall Architecture

This study's application and server work as follows[Fig. 2].

[2] Amazon S3 (Amazon Simple Storage Service) is a cloud-based object storage service supplied by Amazon.

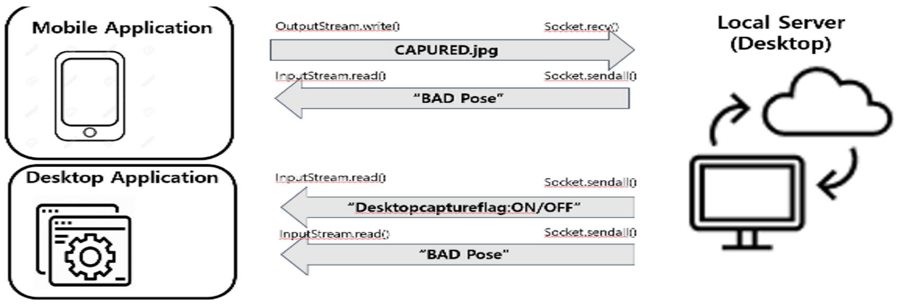


Fig. 2.

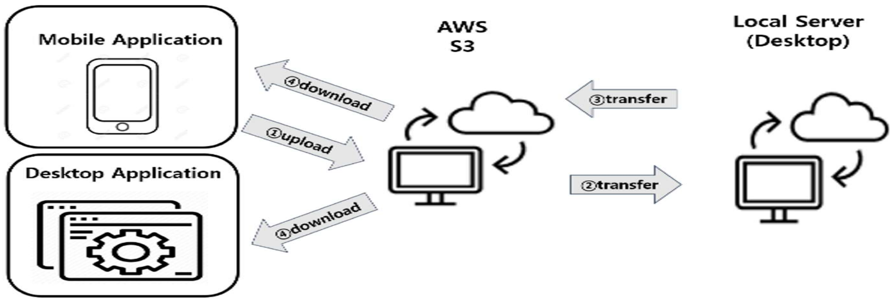


Fig. 3. The system's operation is carried out in the order of the numbers 2-1. We examined direct socket contact with the local server without using S3, as indicated in Figure 2-2, however we discovered that socket communication can be unstable throughout the debugging process. As a result, we picked a secure technique that uses AWS S3.^{2}.

The user's image is sent to the AWS server by the mobile application, and the local server uses it to perform POSE ESTIMATION and posture determination. The local server then sends the pose diagnosis result to the AWS server, which subsequently delivers it to the mobile and desktop applications.

2.2 Core AI Model & Technique

As previously stated, Pose Estimation and Deep Learning are two significant core technologies used in this project.

Human Pose Estimation The difficulty of locating anatomical keypoints or body components in humans is known as pose estimation.[1]

Knowing a person's stance is highly significant in recognizing a person's activity since certain pose movements are usually caused by certain actions of a person. It is separated into two categories: one person/many people and 2D/3D, with 3D and multiple persons regarded more difficult. For human pose estimation, human body modeling is required, and there are three sorts of models.[Fig. 4 5].

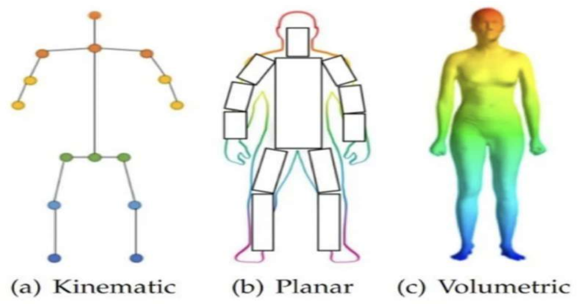


Fig. 4. It's a BODY25 model, which is a type of kinematic model. Starting with the character's neck and moving through the shoulders and arms, legs, and back to the face, each point is numbered.

The following is what we hope to accomplish with pose estimation in this study. In addition, Openpose[2], a well-known open source, was used to do the pose estimate in this work.

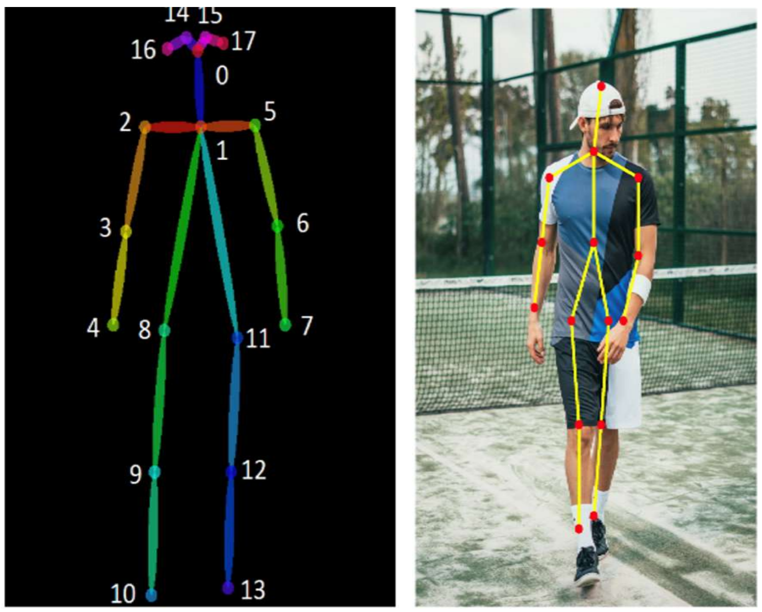


Fig. 5. The BODY25 model, which is a type of kinematic model, is shown on the left. Starting with the character's neck and moving through the shoulders and arms, legs, and back to the face, each point is numbered. The photo on the right was created to be superimposed on the photograph after using the BODY25 model to estimate pose from an actual man image.

Neural Network Model After OpenPose analyzes the keypoints in the user's photo, the pre-trained neural network model determines whether the user's posture is correct. Approximately 30,000 datasets were collected from team members and acquaintances to train the model. In order for the service to function properly, the model must not only be accurate enough but also solve the overfitting problem. First, the coordinate values of keypoints were left alone. In this case, it demonstrated a very high accuracy of 0.9 or higher, but less than 0.5 for images of untrained people. In other words, the problem of overfiitting could not be avoided. Second, the distance between the eyes, ears, and shoulders was calculated and added to the distance between the neck and nose as input data. The accuracy in this case was around 0.8, and the overfitting problem could not be avoided. Finally, we presented the ratios for the four distance values obtained earlier. The distance between the eyes, ears, neck, and nose, divided by the distance between the shoulders, was provided in particular. In this case, the model's accuracy was around 0.8, which was lower than the first method but showed the most freedom from the overfitting problem. The following method was used to assess overfitting and model performance. To begin, the model is trained on 22,000 images of just three team members. The accuracy of the prediction is then tested using photos of five acquaintances. A total of 9000 photos were used, with 1800 photos taken for each person.

Data	Dense Units	Acc on test set	loss	fr0	fr1	fr2	fr3	fr4	batch	epoch
sh ratio	(4, 4)	0.7064	0.543	47	53	49	52	47	16	45
	(4, 8)	0.8076	0.421	62	87	60	81	71	16	45
	(8, 8)	0.7697	0.864	63	93	46	75	71	16	45
	(4, 8)	0.7068	0.545	48	53	48	53	48	32	45
	(8, 8)	0.7206	0.704	43	54	53	67	50	32	45
cor	(4, 8)	0.9054	0.256	53	48	53	53	51	16	45
	(8, 8)	0.979	0.079	59	49	83	56	49	16	45
	(8, 8)	0.9725	0.071	57	48	57	41	56	32	45
dis	(4, 8)	0.7987	0.408	51	53	59	73	56	16	45

Table 1. sh ratio is a model trained after the third method of data processing, cor is a model trained by providing only coordinate values, and dis is a model trained by providing only distance values. The model has two hidden layers, and each pair of Dense Units in the table represents the number of units in the first and second layers. The accuracy is shown when 1/4 of the total 22,000 training data is used as the test set. fr0 to fr4 corresponds to acquaintance number 0 to acquaintance number 4. The value for fr0 represents the accuracy of the trained model's result in determining the posture of acquaintance 0. That is, we assessed the model's performance by examining how much accuracy it represents on data that was not provided during training.

2.3 Application Design

For the design of the mobile application, the left figure of the MVC pattern[Fig. 1] was used. Model, View, and Controller (MVC) is an acronym for

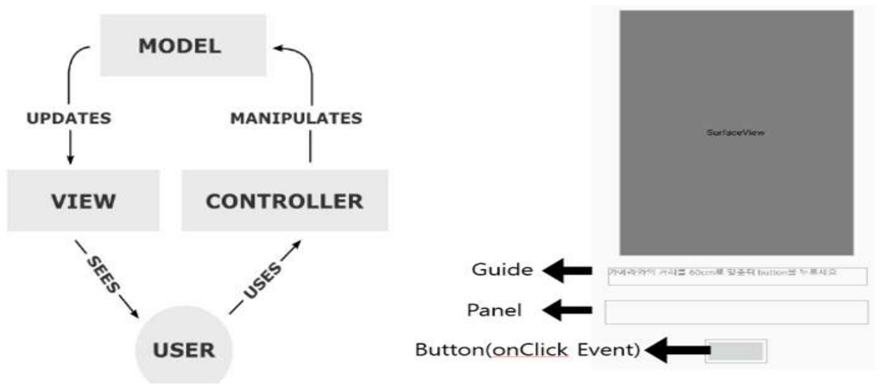


Fig. 6.

Model, View, and Controller. When the user interacts with the controller, the controller obtains data from the model and, using that data, controls the View that is responsible for visual expression and delivers it to the user. The layout of the application is seen in the right figure above[Fig. 6]. It's set up using the MVC approach, and the layout doesn't change much over the application's life cycle.

3 Implementation

The application takes a photo of the user, uploads it to the S3 server, obtains the posture determination result, and sends the user an alarm.

3.1 Camera Setting

Images that disclose human postures are required in order to discriminate human postures, and this must be taught by machine learning. The learning and posture discrimination outcomes of the machine learning model may vary significantly at this point, depending on the angle and distance at which the camera captures the subject. To put it another way
"It is feasible to determine the posture of diverse users since the model is trained with the image. Additional variables such as the distance to the camera and the user's sitting height should be taken into account. Aside from the learnt image, the user's seating posture may change."[3],

This has been discovered in earlier research that are similar to this one. As a result, in this study, [Fig. 7]the camera configuration work is carried out in order to fix the shooting angle and distance so that it becomes the state shown in the application's initial execution stage.
1) [Fig. 8]Set the camera's distance from the subject's face to roughly 60cm.
2) [Fig. 9]Set the camera to a 60-degree angle with the subject's face.

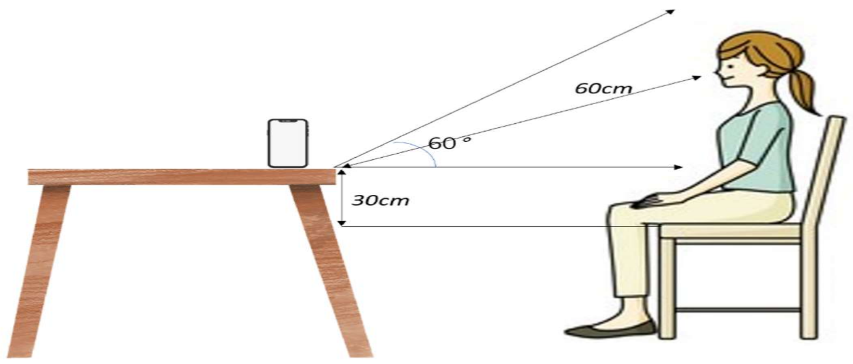


Fig. 7. It is OK to set the angle to 10 degrees or higher at this time if the height of the ground where the camera is positioned is too high or too low for a person to enter the screen. A 30 cm difference in height between the chair and the desk is ideal.

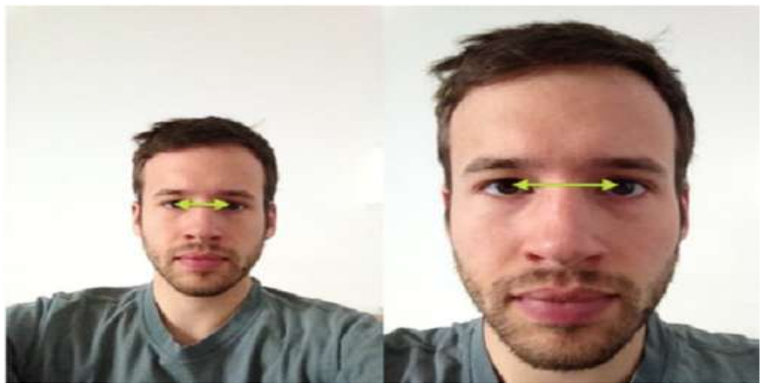


Fig. 8. To determine the distance between the camera and the face, use the distance between the eyes. We made use of open source software.[4]

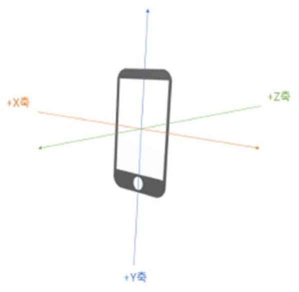
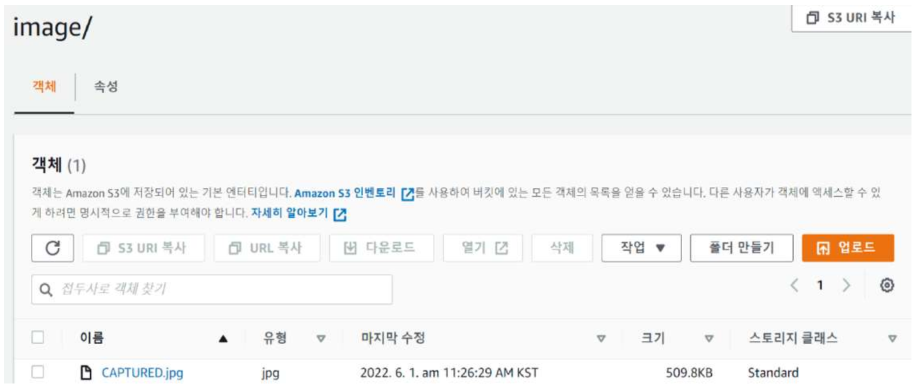


Fig. 9. The accelerometer sensor included inside the smartphone is used to measure the angle created by the phone. We made use of open source software.[5]

3.2 Image Capture & Upload

Following the completion of the camera settings, the front camera's screen is collected every 10 seconds and transferred to the AWS S3 server.



객체 (1)

객체는 Amazon S3에 저장되어 있는 기본 엔터티입니다. Amazon S3 인벤토리 를 사용하여 버킷에 있는 모든 객체의 목록을 얻을 수 있습니다. 다른 사용자가 객체에 액세스할 수 있게 하려면 명시적으로 권한을 부여해야 합니다. 자세히 알아보기

이름	유형	마지막 수정	크기	스토리지 클래스
CAPTURED.jpg	jpg	2022. 6. 1. am 11:26:29 AM KST	509.8KB	Standard

Fig. 10. The captured photo is uploaded to S3

3.3 Pose Diagnosis Result Download & Alarm

On the local server, photos collected in S3 are subjected to POSE DIAGNOSIS, with the findings being posted back to S3 in the form of txt. Only when no one shows or when there is a bad pose is the result downloaded and an alarm displayed on the screen in the mobile application. The cell phone is also configured to vibrate in the event of BAD POSE. [Fig. 12] Each case is shown.

4 Limitations and Discussions

Fixed Camera Distance & Angle To assure the performance of machine learning, the distance and angle between the camera and the subject had to be fixed in this investigation. In other words, because the smartphone must be used in a fixed location in front of the user, it may be troublesome for the user who has to use the smartphone while working.

Low AI Model Performance Because the AI model developed in this study's performance, i.e. its accuracy, is not particularly high, just the attitude determination of the team members who conducted this study is possible at this time. Datasets play a big role in this. It was difficult to gather enough data for many people since the image data required for learning had to be taken from a set angle and distance.

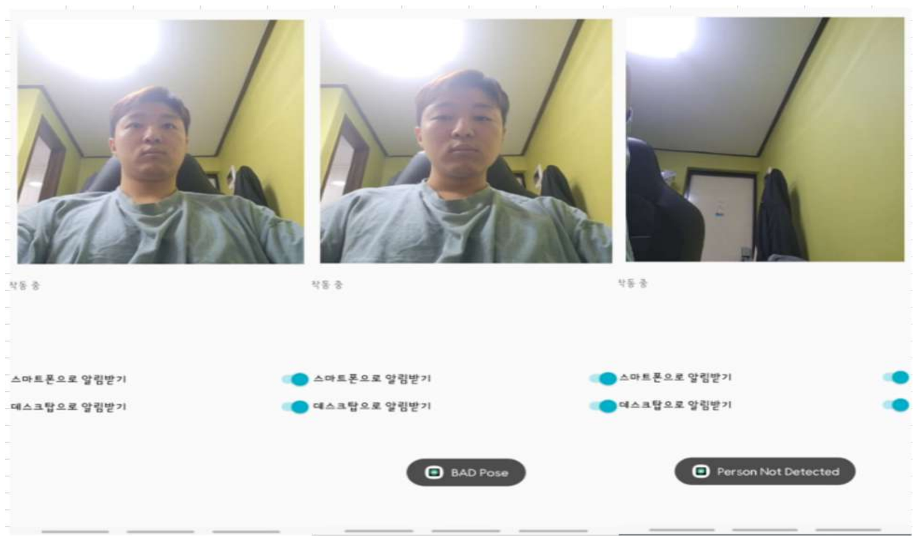


Fig. 11.
1)Good Pose: no notification pops up
2)Bad Pose: vibrate with alarm
3)Person Not Detected:display only notifications
The on/off switch on the screen is an alarm ON/OFF feature, allowing you to set or disable the alarm in mobile and desktop apps.

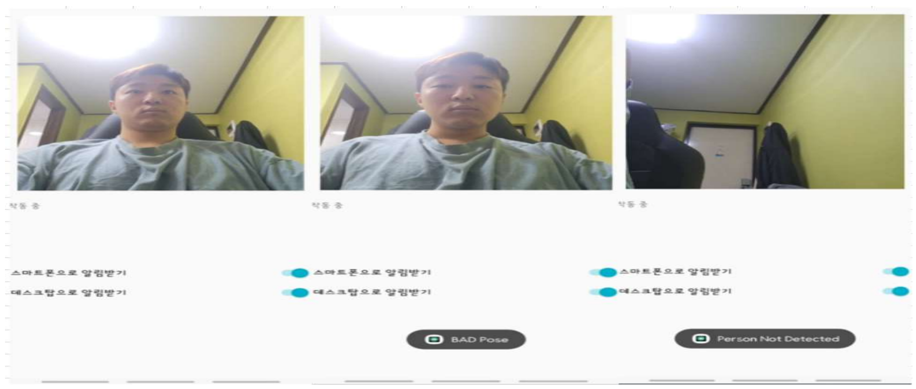


Fig. 12. If you're using a desktop application, the following notification appears only if you're in a bad pose..

Syncronization Between Server and Actual Pose There is a 5-10 second delay once a person's pose is altered before the mobile application detects it and displays a new notification. This is due to a lack of hardware resources, and it appears that a powerful graphics card will address the problem.

Local Server Because posture determination is done on the local server in this study, there is the drawback that operation is only possible when the desktop is running. Because it is expensive, it was not possible to establish a cloud server and arrange it such that pose determination could be performed at all times.

5 Roles

This study was conducted with two people due to the absence of two of the four team members that started at the beginning.

Application(Mobile & Desktop) : Jongwon
- Camera Distance & Angle Setting
- Periodic (10s) Capture
- Communication with Server
- Show (bad) Pose Determination & Make Alarm

Machine Learning : Dongwoo
- Training
- Improving Model Performance

Server : Dongwoo,Jongwon
- Pose Estimation (with OpenPose)
- Communication with Application
- Receive : Captured Image from Application
- Send : Pose Determination Result

Acquiring & Preprocessing Dataset : Dongwoo,Jongwon
- Make Dataset
- Extract image from video

References

1. Zhe Cao, Tomas Simon, Shih-En Wei, Yaser Sheikh; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7291-7299
2. https://github.com/CMU-Perceptual-Computing-Lab/openpose
3. Han, J.-Y., Park, J.-H. (2020). Turtle Neck Syndrome Posture Correction Service Using CNN-based Learning Model. The Journal of the Korea Contents Association, 20(7), 47–55. https://doi.org/10.5392/JKCA.2020.20.07.047

4. I. König, P. Beau and K. David, "A new context: Screen to face distance," 2014 8th International Symposium on Medical Information and Communication Technology (ISMICT), 2014, pp. 1-5, doi: 10.1109/ISMICT.2014.6825217. https://github.com/philiiiiiipp/Android-Screen-to-Face-Distance-Measurement
5. https://github.com/dnjswhddnjs/user.activity