Monitoring the risk of COVID-19

Eunbin Kang Donghyun Kim Jungrae Kim Seonkyo Ok
Seoul National University, Seoul, Korea
{eunbin1227, dh047, tyg03485, timosk}@snu.ac.kr

Abstract

Indicates the risk level of COVID-19 and monitors it in real time to prevent mutual infection in advance. The most important factor in COVID-19 infection is whether a mask in worn, which consists of two methods: for identifying and detecting location of the eyes, nose and mouth, and to detect the characteristics of the mask using the OpenCV library. In addition, when the mask is not worn, it detects the opening of the mouth or conversation to show the risks. Code is available at https://github.com/uyw4687/mmc.git.

1. Introduction

In 2020, humankind has been badly hit by the virus called COVID-19, or SARS-CoV-2, which causes respiratory illnesses. From the beginning of 2020, it has been spread all over the world and still it persists around us. Since this virus is transmitted by tiny droplets that occur while coughing, sneezing and speaking, it is commonly recommended to wear the mask in public places. However, lots of people do not properly comply with these measures.

Without any safety rules, the number of infections grow exponentially, which can be described with the transmission rate, R0. And wearing a mask and keeping social distances effectively help lowering the value. So, in this project, given the images of people on streets, we're going to determine if masks are worn properly or not and divide the result into three classes. For example, if the nose and the mouth are covered, it is proper. If one of them are exposed, it's improper. If no masks are found, it's also improper and divided into yet another class. In addition, we will figure out whether he/she opens his/her mouths or talk to someone to measure additional risks.

Consequently, the goal is to monitor masked faces and detecting opened mouth or conversation at once to express the COVID-19 Risk Index as 0-100. (100 is the highest risk)

2. Background

2.1. Histogram of Oriented Gradients (HOG)

The histogram of oriented gradients (HOG) is a feature descriptor for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. This method is computed on a dense grid of uniformly spaced cells and uses overlapping local contrast normalization for improved accuracy.

The essential thought behind the histogram of oriented gradients descriptor is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. The image is divided into small connected regions called cells, and for the pixels within each cell, a histogram of gradient directions is compiled. The descriptor is the concatenation of these histograms. For improved accuracy, the local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image, called a block, and then using this value to normalize all cells within the block. This normalization results in better invariance to changes in illumination and shadowing.

2.2. Gaussian Filter

Gaussian filter is a filter used to blur images. This method can blur the image by placing weights on the surrounding pixels, and the value of Gaussian filter in the two-dimensional image is as follows.

$$g(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

2.3. Sobel filter

The Sobel filter is a filter used to detect the gradient value of a pixel. For two-dimensional images, there are x- and y-directional filters, both of which are as follows.

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

2.4. Canny Edge Detection

Canny edge detector is a method of detecting the lines in the image. First, apply Gaussian filter to eliminate noise. Then get the gradient in each direction through the Sobel kernel. The edge gradient for each pixel is as follows.

$$Gradient(G) = \sqrt{(G_x)^2 + (G_y)^2}$$

$$Angle(\theta) = \tan^{-1}(\frac{G_y}{G_x})$$

After that, remove pixels that do not make edges by scanning all pixels. Finally, detect edges by holding the appropriate threshold.

3. Related Work

Since prior face detection algorithms such as Adaboost [1] use approaches like deep learning in that it trains a model for detection, we firstly considered using existing deep learning models [2] to detect faces. However, the result of Adaboost was quite good enough so we used both methods to implement the algorithm using the existing models and methods. The MaskedFace-Net [3] dataset where it provides datasets of properly or improperly masked faces also implies methods to detect proper or improper mask wearing. To make the dataset, it used a facial point annotation model trained on 300-W dataset [4][5][6].

By using facial annotations, we can determine if masks are properly worn or not. In addition, there is also a existing study [7] where it uses deep-learning based approach to determine if masks are worn or not. We will refer to the results from this to assess our approach [8][9][10][11].

4. Methodology

In our methodology, the person on the camera has a status of one of the tree's child nodes. (Total four status)

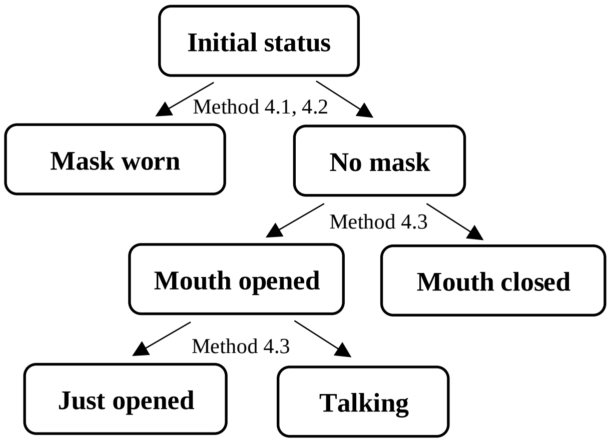

Figure 1. Status

4.1. Detecting faces, 68 landmarks, and face parts

In this part, the program checks if a mask is properly worn or not. If mask is worn but the nose is exposed, then it's improper. If there's nothing that covers noses and mouths, it's also improper. So, the only proper case where the program outputs 'Good' instead of 'Alert!' is when a face is covered with something so that no noses or mouths are detected.

Firstly, as in the Figure 2.1.1, faces are detected and based on the result, the 68 landmark deep learning model [6] is run. A face without a mask is detected in this way in fast and precise manner. Since a face can be detected even with a mask, the program checks if the nose part is detected as a nose by using the cascade classifier [1] in the position of nose. Most of the time, if a mask is worn, no faces are detected using the previous face detector which makes use of HOG features. Then, it mainly uses the result of cascade classifiers to find eyes, noses, and mouths.

In this part, naïve implementation will make the program slow. So, each classifier must be run when needed and only on the pixels where applicable. If no faces are found, then we don't find faces again with the cascade classifier though since it is quite probable that it would also not find a face when a mask is worn. So, for performance, it finds eyes. The ideal case is when two eyes are found. Then we can remove false findings of mouths and noses so that we can make results. Noises are removed by finding the distances between two eyes and finding the distance between the face parts so that abnormally distanced findings are ignored.

Also, the position of noses and mouths are checked by invalidating any findings of them that are far away from the line which is perpendicular to the line segment that connects the eyes and originates from the middle of the line segment. The code itself is PyLint checked and modularized enough for readability and future usability.

4.2. Detecting features of a mask

This part is mainly detection of a mask. Additionally, it checks the mask is worn correctly or the mask is just laid.

First, using a function that recognizes adjacent pixels as a single contour, the program finds candidates of a mask. These contours are partitioned by canny edges. Among the candidates, it chooses the closest to masks based on color and shape. This algorithm recognizes mask color on HSV values because most masks have solid colors. And it uses only the simple feature that a mask is close to a hexagon or a circle.

An image with correctly worn mask always has two eyes. If a side face, it has one eye. This algorithm determines by relative position of the eye and the mask.

Of course, this algorithm supports color masks as well as white/black masks. However, there is a disadvantage that it cannot detect a mask on which something is drawn like fashion mask.

4.3. Detecting the movement of a mouth

In this part, the algorithm tries to detect whether he or she is speaking. Based on the 68 landmark deep learning model [6], we could calculate shape and position of mouth. So, If the history of lip distance between upside and downside changed a lot as time frame, we can say, “he or she is speaking”. Of course, They just open the mouth and not move. So, we must calculate change history of lip distance with variance or others.

As a result, the program detects the speaking motion and some stuff like just opening the mouth. Also, it prints message for previous motion like “Please, close the mouth for COVID-19”, “Please, stop speaking”.

5. Experiment

5.1. Detecting faces, 68 landmarks, and face parts

The mask detection results were as follows. If you don't wear a mask, as you can see in Figure 2.1.1., you can spot points on your face to locate the eyebrows of your eyes, nose, mouth, and eyebrows, recognize your face and nose, and mark the middle of your nose. This was used to recognize when the mask was worn and not used in Figure 2.1.2. and Figure 2.1.3. Not only is it not just wearing a mask, but it also recognizes the nose and not properly wears a mask.

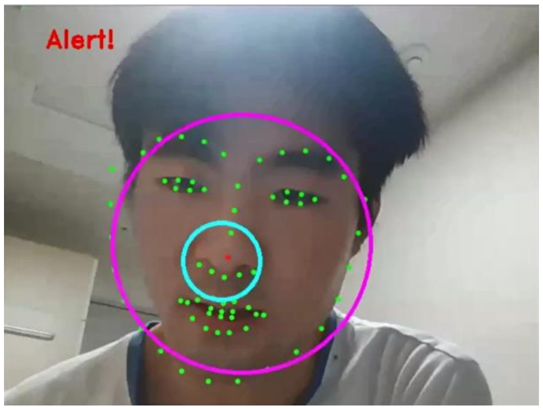

Figure 2.1.1. Without a mask

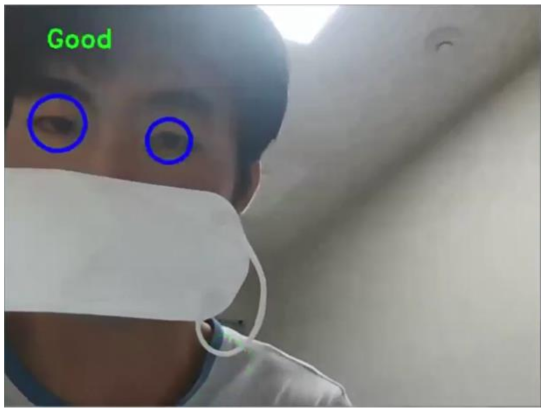

Figure 2.1.2. Mask Properly Worn

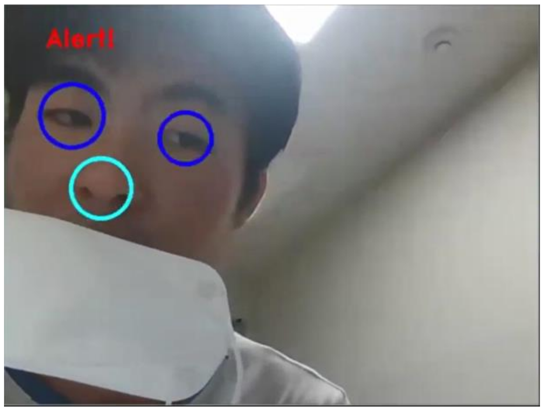

Figure 2.1.3. Nose detected(improper)

5.2. Detecting features of a mask

Detection results using the characteristics of the mask are shown in Figure 2.2.1. and Figure 2.2.2. Both photos were normally aware of the mask, and regardless of the shape or color of the mask, if it is in the shape of the mask, it can be confirmed that both are aware of it. In addition, the mask can be recognized on the side, so it can be used more as a general purpose.

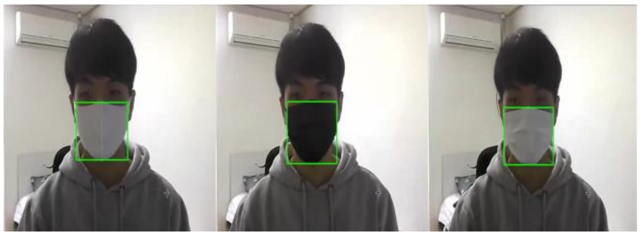

Figure 2.2.1. Detecting various masks

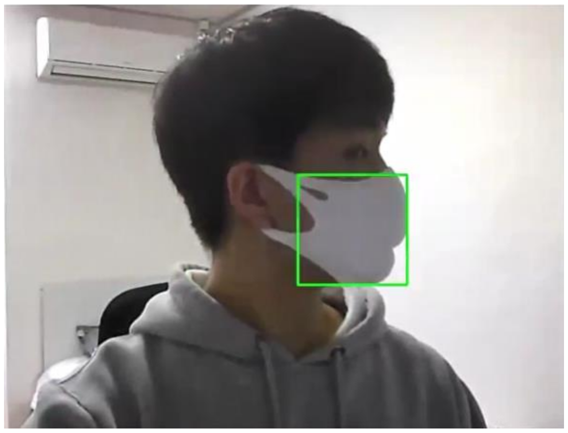

Figure 2.2.2. Detecting sides

5.3. Detecting the movement of a mouth

The results of detecting mouth shape are shown in Figure 2.3.1 and 2.3.2 and 2.3.3. Each recognizes the distance between the two lips, and when the two lips are attached, no notice is given, as in Figure 2.3.1. On the other hand, when the distance between the two lips is not zero, you can see a notice to close the mouth, as shown in Figure 2.3.2. Furthermore, if the lips move continuously, you can see a notice telling you not to talk, as you can see in Figure 2.3.3.

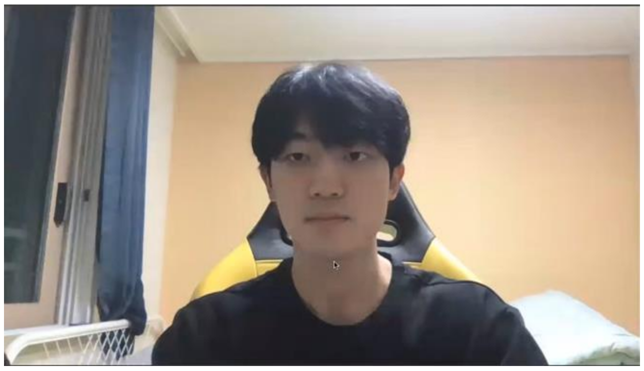

Figure 2.3.1. Mouth Closed

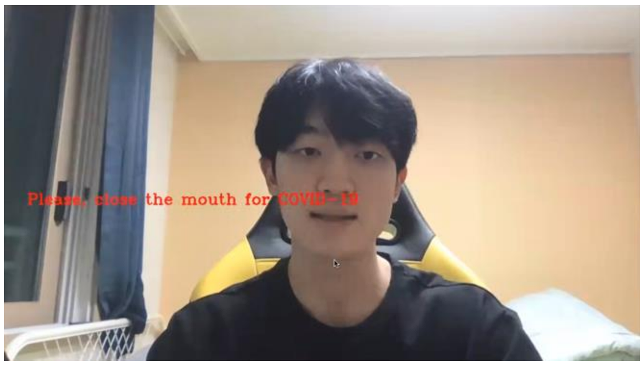

Figure 2.3.2. Mouth Opened

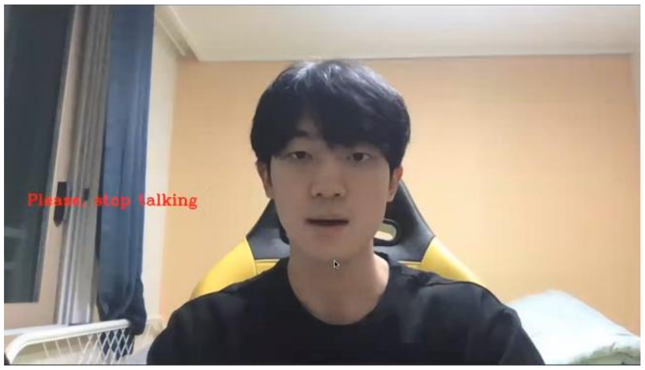

Figure 2.3.3. Talking

6. Conclusion

6.1. Mask detection

In the case of mask detection, the experiment was carried out in two ways. Both methods were able to recognize the mask successfully in general cases. However, the method of finding a mask by recognizing the eyes, nose, and mouth was difficult to detect because the number and appearance of the eyes, nose, and mouth in the case of profile changes. (False negative) There was also room for recognition of the lung curve corresponding to the mask and other objects with similar characteristics to recognize the mask based on that. (False positive) Therefore, using both methods, we could more accurately determine whether a person in the camera is wearing a mask.

6.2. Mouth detection

When the mouth was not wearing a mask, both when the mouth was closed, when the mouth was open, and when the mouth was constantly moving.

However, the limitation is that there is a slight delay in having a conversation because it is necessary to grasp the constant movement of the mouth.

But by detecting this movement of the mouth, the risk could be observed with more varied status.

6.3. Result

To sum up, after analyzing a person's status in this case, we can calculate the risk of exposure to COVID-19 for each individual. So, in conclusion, we're going to scale the risk from 0 to 100. For this purpose, m1 (if wearing 0, not wearing 1), m2 (if not open 0, open 1), m3 (if not talking 0, talking 1) can be used to indicate the risk level from 0 to 100 using the formula below.

$$Risk = m_1 * \left(0.6 + m_2 * (0.2 + 0.2 * m_3)\right) * 100$$

The formula above has a zero risk when wearing a mask. However, if the mask is not worn, the risk is increased by 60; if the mouth is open, the risk is increased by 20; if the conversation is in progress, the risk is increased by 20. Therefore, we can see how much COVID-19 exposure has been made in real time through the corresponding calculation. Of course, the weights may be somewhat inaccurate because they are set at random, but it would be meaningful if they contribute to the prevention of corona by informing the screen viewers that there is a risk.

6.4. Future Works

This result show classical methods are viable to implement the necessary functionalities. To improve the result, if possible, the nose cascade classifier should be trained more to detect more precisely. Also, the algorithm itself may be heuristically more improved to fasten the procedures. Though this program runs fast enough, there still is a room for optimization.

Also, there's a point that this kind of traditional methods are easy to use and sometimes fast enough to compare to deep learning. So, this work suggests two different possibilities. Properly using traditional methods to solve real-world problems and making deep learning procedures lighter and easier. Either way, the progress will be impressive enough.

References

[1] Paul Viola, and Michael Jones. Rapid Object Detection using a Boosted Cascade of Simple Features. In *CVPR*, 2001.

[2] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *arXiv preprint arXiv: 1604.02878*, 2016.

[3] Adnane Cabani, Karim Hammoudi, Halim Benhabiles, and Mahmoud Melkemi. MaskedFace-Net - A Dataset of Correctly/Incorrectly Masked Face Images in the Context of COVID-19. *arXiv preprint arXiv:2008.08016*, 2020.

[4] Christos Sagonas, Epameinondas Antonakos, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 300 Faces In-The-Wild Challenge: Database and results. *Image and Vision Computing, Special Issue on Facial Landmark Localisation "In-The-Wild"*, 2016.

[5] https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/

[6] https://github.com/davisking/dlib-models#shape_predictor_68_face_landmarksdatbz2

[7] Biparnak Roy, Subhadip Nandy, Debojit Ghosh, Debarghya Dutta, Pritam Biswas, and Tamodip Das. MOXA: A Deep Learning Based Unmanned Approach For Real-Time Monitoring of People Wearing Medical Masks. *Transactions of the Indian National Academy of Engineering*, 2020.

[8] https://www.nationalgeographic.com/science/2020/09/face-mask-recognition-has-arrived-for-coronavirus-better-or-worse-cvd/

[9] https://tryolabs.com/blog/2020/07/09/face-mask-detection-in-street-camera-video-streams-using-ai-behind-the-curtain/

[10] https://machinelearningmastery.com/how-to-perform-face-detection-with-classical-and-deep-learning-methods-in-python-with-keras/

[11] http://www.riss.kr/search/detail/DetailView.do?p_mat_type=be54d9b8bc7cdb09&control_no=29833f32aaaf9e53ffe0bdc3ef48d419&outLink=K