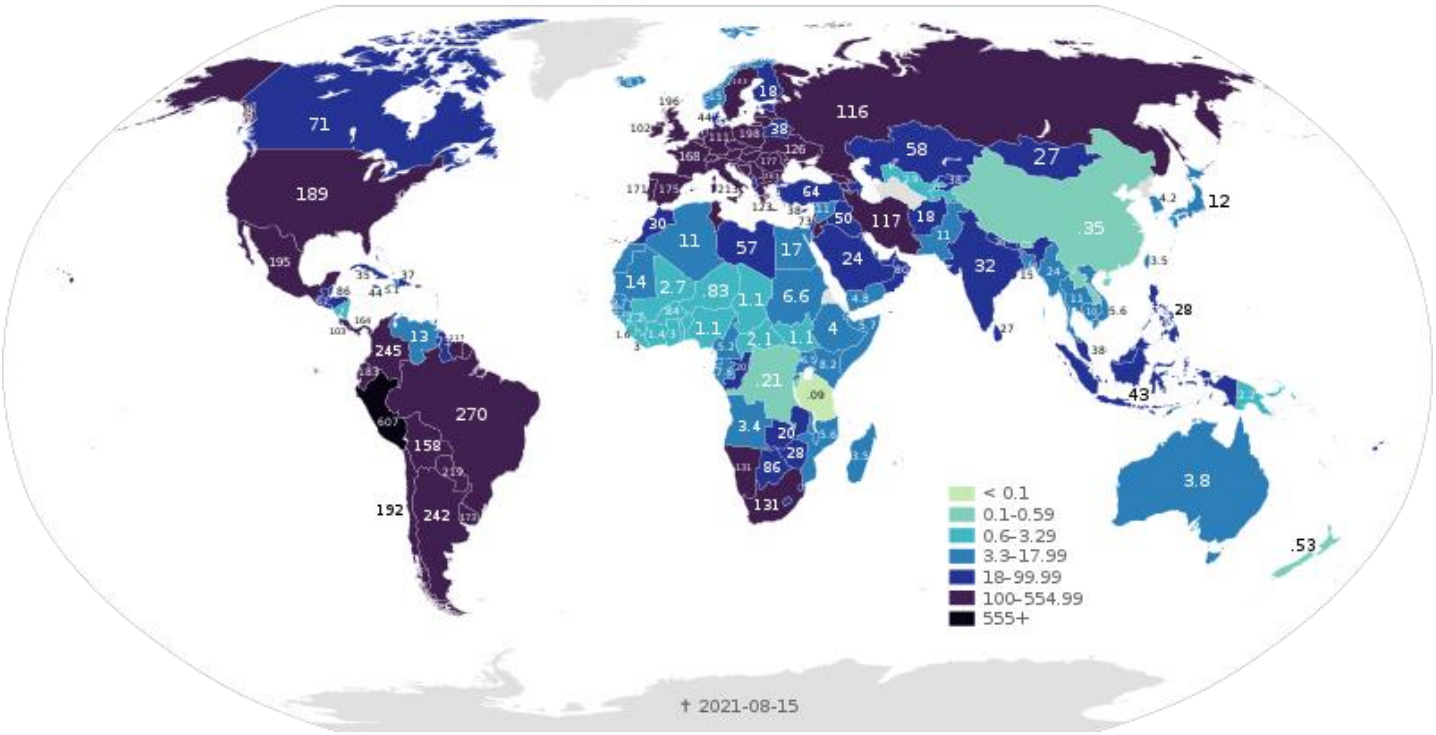
1. Introduction

The outbreak of coronavirus disease (COVID-19) has been declared by Public Health Emergency of International Concern (PHEIC) and the virus has now spread to many countries and territories. While a lot of contact with respiratory droplets of an infected person (generated through coughing and sneezing). Individuals can also be infected from touching surfaces contaminated with the virus and touching their face. While Covid-19 continues to spread, is it important that communities take action to prevent further transmission, reduce the impacts of the outbreak and support control measures. The image would give a map of cases of Covid-19 in the world:

COVID-19 Outbreak World Map Total Deaths per Capita from Template:2019–20 coronavirus pandemic data and List of countries and dependencies by population using code at the talk page of the file on Commons.



COVID-19 Outbreak World Map Total Deaths per Capita, 16 August 2021(Source: Wikipedia, Covid-19 pandemic by country and territory)

The image presented on page are based on reported cases and deaths. While in several developed countries the ratio of total estimated cases and deaths is low and close to 1, for some countries it may be more than 10 or even more than 100. There is one good news though, the fact that pandemic Covid-19 is preventable. Prevent Covid-19 and help end the pandemic like wear a mask, wash hands, maintain safe distance & limit mobility, and get vaccinated. The aim of this project is to research the pandemic Covid-19 problem and come up with a general solution accessible to public places.

1.1. Problem Statement

Coronavirus disease (COVID-19) is an infectious disease caused by the SARS-CoV-2 virus. Most people who fall sick with Covid-19 will experience mild to moderate symptoms and recover without special treatment. However, some will become seriously ill and require medical attention.

The virus can spread from an infected person's mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe. These particles range from larger respiratory droplets to smaller aerosols. We can be infected virus if we are near someone who has Covid-19, or by touching a contaminated surface and then your eyes, nose or mouth. The virus spreads more than easily indoors and in crowded areas.

Since Covid-19 is a chronic disease, preventing the spreading of virus is an absolute necessity which can be started by several steps that we can take every day. Some of these methods can be followed:

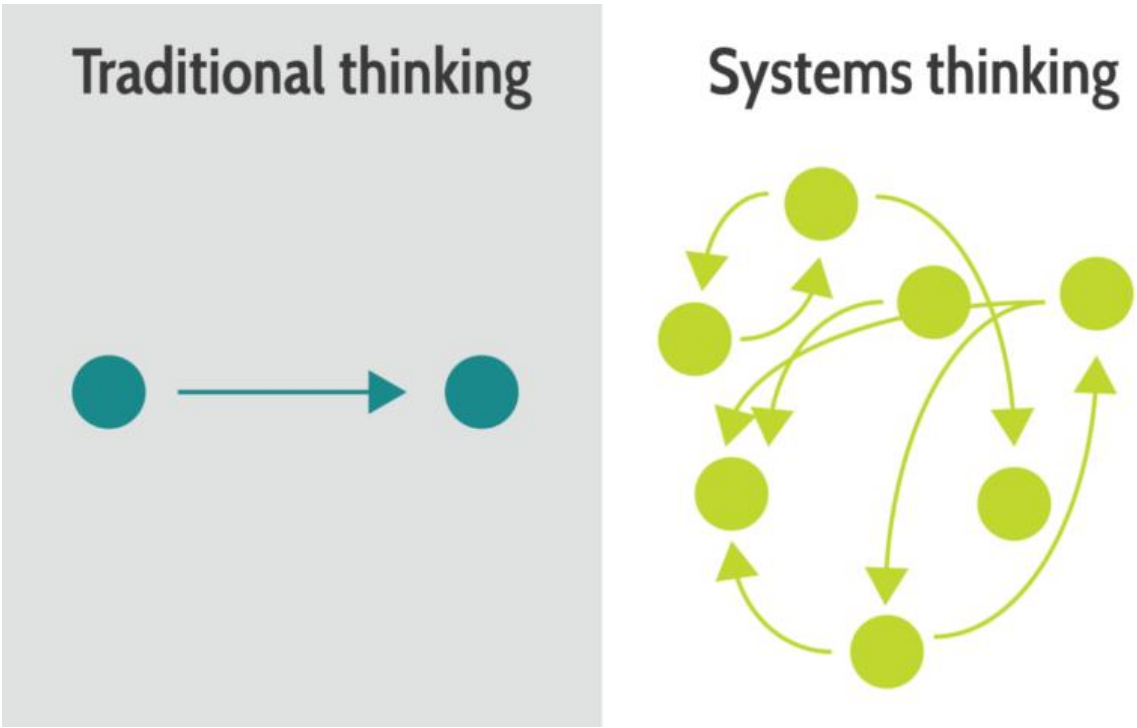
- Maintain a safe distance from others, even if they don't appear to be sick.
- Wear a mask in public, especially indoors or when physical distancing is not possible.
- Choose open, well-ventilated spaces over closed ones. Open a window if indoors.
- Clean your hands often. Use soap and water, or an alcohol-based hand rub.
- Get vaccinated when it's your turn. Follow local guidance about vaccination.
- Cover your nose and mouth with your bent, elbow or a tissue when you cough or sneeze.
- Stay home if you fell unwell.

Here the focus is on the physical distancing part and keeping track of it. Human visual recognition is predicting use or not use a mask by a person and often involves domain knowledge and processing methods to obtain features from object detection to build correct machine learning models. Covid-19 is a major problem these days, mainly the elderly. Covid-19 can be caused by lack of self safety and self-awareness or even lack of physical distancing. So, one way of controlling Covid-19 can be checking by self safety (e.g. wearing a mask).

2. Methodology

2.1. System Thinking

A system is made up of a group of interconnected components. System thinking is a holistic way of thinking and demonstrates how different pieces of jigsaw puzzle come together to form beautiful pictures. Changing one part of a system may affect other parts or the whole system. It may be possible to predict these changes in patterns of behavior. Now, we will take a look at the whole system of the Covid-19 problem.



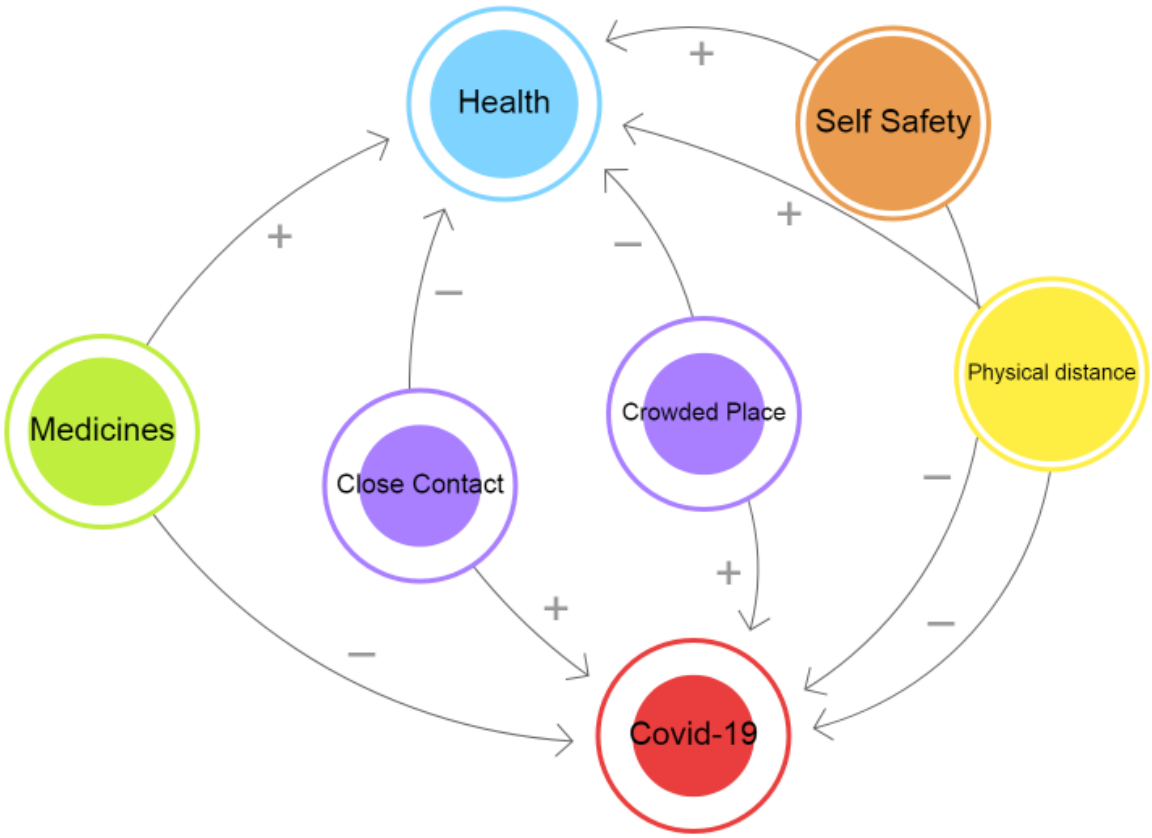
Traditional thinking vs system thinking (Source: *https://kindling.xyz/futures/systems-thinking/)*

We can apply the system thinking process to our Covid-19 problem, in fact. By breaking the system into components and dividing relations between them, we can come up with a systems map. The components of our system are as follows:

- Health
- Physical distance
- Self safety
- Covid-19
- Crowded place
- Close contact other people, and
- Medicines

All of these components are interconnected to each other and affect each other vastly. For instance, self safety positively impacts health, physical distance and medicines more the positive benefits to health. Close contact with other people would negatively affect our health since components of close contact do not often fall into preventing the spread of the virus. Alternatively, close contact with other people would positively affect our chances of being spread Covid-19.

Here we present a systems diagram of the problem at hand:



Here the leverage point is the Self Safety as it can be easily checked via mask detector device. So, a system would be devised to check whether people wear masks or not.

2.2. Approach

Almost everyone knows public places have security cameras inside that collect huge amounts of data. AI can help us recognize humans easily from such data. Human Visual Recognition dataset was built from the images of people using mask and non-use mask from capture by openCV and training model with Convolution Neural Network Algorithm.

The experiments have been carried out with some people scanning their faces before entering a mall, office, etc. Each person wears a mask and some haven't wearing mask.

The aim is to build a machine learning model that would divide the classification into 2 groups: Images people wear a mask and people not wear a mask and some don't.

2.3. Design Thinking

Design thinking is an iterative process in which we seek to understand the user, challenge assumptions, and redefine problems in an attempt to identify alternative strategies and solutions that might not be instantly apparent with our initial level of understanding. Design thinking can help us understand how designers' work processes can help systematically extract, teach, learn and apply these human-centered techniques to solve problems in a creative and innovative way in our designs, in our businesses, in our countries, in our lives. Here we will see how we can design a system that actually keeps track of human visual recognition and links it to reducing Covid-19. Since we have devised our system map, we can proceed to design thinking to empathize with the people we are building the solution for, define the problem clearly, come up with ideas and a prototype and then test it among users. We now proceed with the steps of design thinking.

Empathy: Covid-19 is a major problem in different age groups infants, youth, middle aged and the erderly causing millions of deaths since 2019 year.

Define : In order to tackle Covid-19, the easiest ways are to control physical distance or social distancing and visual recognition for self safety. Human visual recognition need to be carefully, know the difference in data classification.

Ideate: How the problem can be solved with or without AI

Prototype: Choose feasible ideas from the pool ideas and select the ones that can be tested with real people. Get people data from capturing images by opencv and training model with watson studio and do simple visualise data in python.

Test: Get some quick comments about problems identified in the prototype phase and come up with a full fledged solution in a jupyter notebook.

3. Implementation

Almost everyone knows public place have security cameras inside that collect huge amounts of data. AI can help us recognize humans easily from such data. The dataset we are using with capturing images and mostly from (https://github.com/prajnasb/observations/tree/master/experiements/data).

Here we take a dataset containing information about the images of people with masks and without masks. A simple CNN classification algorithm would be used to divide classification into 2 groups: Images people wear a mask and people not wear a mask. And the code is divided into 3 major parts.

The first part is Data Pre-Processing. Here we create 2 folders for the dataset ("with mask" and "without mask"), pull images, gray the images, resize the gray images to 100 x 100 scale.

DATA PRE-PROCESSING

In [1]:
```
# step 1 - import library
import cv2,os

data_path=r'dataset'
categories=os.listdir(data_path)
labels=[i for i in range(len(categories))]

label_dict=dict(zip(categories,labels)) #empty dictionary

print(label_dict)
print(categories)
print(labels)
```

```
{'with mask': 0, 'without mask': 1}
['with mask', 'without mask']
[0, 1]
```

In [2]:
```
# step 2 - import 2 folders for datasets("mask", "no mask")
img_size=100
data=[]
target=[]


for category in categories:
    folder_path=os.path.join(data_path,category)
    img_names=os.listdir(folder_path)

    for img_name in img_names:
        img_path=os.path.join(folder_path,img_name)
        img=cv2.imread(img_path)

        try:
            gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
            #Coverting the image into gray scale
            resized=cv2.resize(gray,(img_size,img_size))
            #resizing the gray scale into 50x50, since we need a fixed common size for all the images in the dataset
            data.append(resized)
            target.append(label_dict[category])
            #appending the image and the label(categorized) into the list (dataset)

        except Exception as e:
            print('Exception:',e)
            #if any exception rasied, the exception will be printed here. And pass to the next image
```

```
In [3]: import numpy as np

        data=np.array(data)/255.0
        data=np.reshape(data,(data.shape[0],img_size,img_size,1))
        target=np.array(target)

        from keras.utils import np_utils
        import tensorflow as tf
        tf.get_default_graph

        new_target=np_utils.to_categorical(target)

        Using TensorFlow backend.

In [4]: np.save('data',data)
        np.save('target',new_target)
```

Save the images and paths as data.npy and target.npy

The second part is Convolutional Neural Network. Now we set up our CNN model. First load the processed images and paths.

Convolutional Neural Network Architecture

```
In [5]: import numpy as np

        data=np.load(r'data.npy')
        target=np.load(r'target.npy')

        import tensorflow as tf
        tf.get_default_graph

Out[5]: <function tensorflow.python.framework.ops.get_default_graph()>
```

```
from sklearn.model_selection import train_test_split

train_data,test_data,train_target,test_target=train_test_split(data,target,test_size=0.1)
```

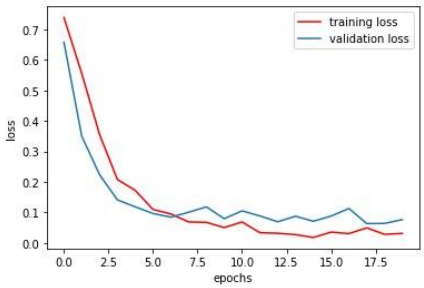
Split our data (processed images)

```
checkpoint = ModelCheckpoint('model-{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True,mode='auto')
history=model.fit(train_data,train_target,epochs=20,callbacks=[checkpoint],validation_split=0.2)
```

```
WARNING:tensorflow:From C:\Users\Achmad Reza\Anaconda3\lib\site-packages\keras\backend\tensorflow_backend.py:422: The name tf.g
lobal_variables is deprecated. Please use tf.compat.v1.global_variables instead.

Train on 908 samples, validate on 227 samples
Epoch 1/20
908/908 [==============================] - 78s 86ms/step - loss: 0.7392 - accuracy: 0.5463 - val_loss: 0.6578 - val_accuracy:
0.7577
Epoch 2/20
908/908 [==============================] - 84s 92ms/step - loss: 0.5554 - accuracy: 0.7236 - val_loss: 0.3499 - val_accuracy:
0.8899
Epoch 3/20
908/908 [==============================] - 181s 199ms/step - loss: 0.3552 - accuracy: 0.8447 - val_loss: 0.2235 - val_accuracy:
0.9295
Epoch 4/20
908/908 [==============================] - 252s 278ms/step - loss: 0.2078 - accuracy: 0.9207 - val_loss: 0.1408 - val_accuracy:
0.9559
Epoch 5/20
908/908 [==============================] - 258s 284ms/step - loss: 0.1721 - accuracy: 0.9339 - val_loss: 0.1181 - val_accuracy:
0.9471
Epoch 6/20
908/908 [==============================] - 259s 285ms/step - loss: 0.1094 - accuracy: 0.9604 - val_loss: 0.0962 - val_accuracy:
0.9559
Epoch 7/20
908/908 [==============================] - 255s 281ms/step - loss: 0.0946 - accuracy: 0.9648 - val_loss: 0.0842 - val_accuracy:
0.9515
```

Creating and save model

In [10]:
```
# visualization for train model

from matplotlib import pyplot as plt

# loss visualise
plt.plot(history.history['loss'],'r',label='training loss')
plt.plot(history.history['val_loss'],label='validation loss')
plt.xlabel('epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```



The last part is Mask Detection. After training model and has already to applied camera. Select the model which has the highest accuracy. Next load the haarcascade classifier for detecting the face. And then started the video capture.

Detection

In [12]:
```
from keras.models import load_model
import cv2
import numpy as np
import tensorflow as tf
tf.get_default_graph
```

Out[12]:
```
<function tensorflow.python.framework.ops.get_default_graph()>
```

In [15]:
```
model = load_model(r'model-018.model')

face_clsfr=cv2.CascadeClassifier(r'haarcascade_frontalface_default.xml')

source=cv2.VideoCapture(2)

labels_dict={0:'Sip',1:'PAKAI MASKER!'}
color_dict={0:(0,255,0),1:(0,0,255)}
```

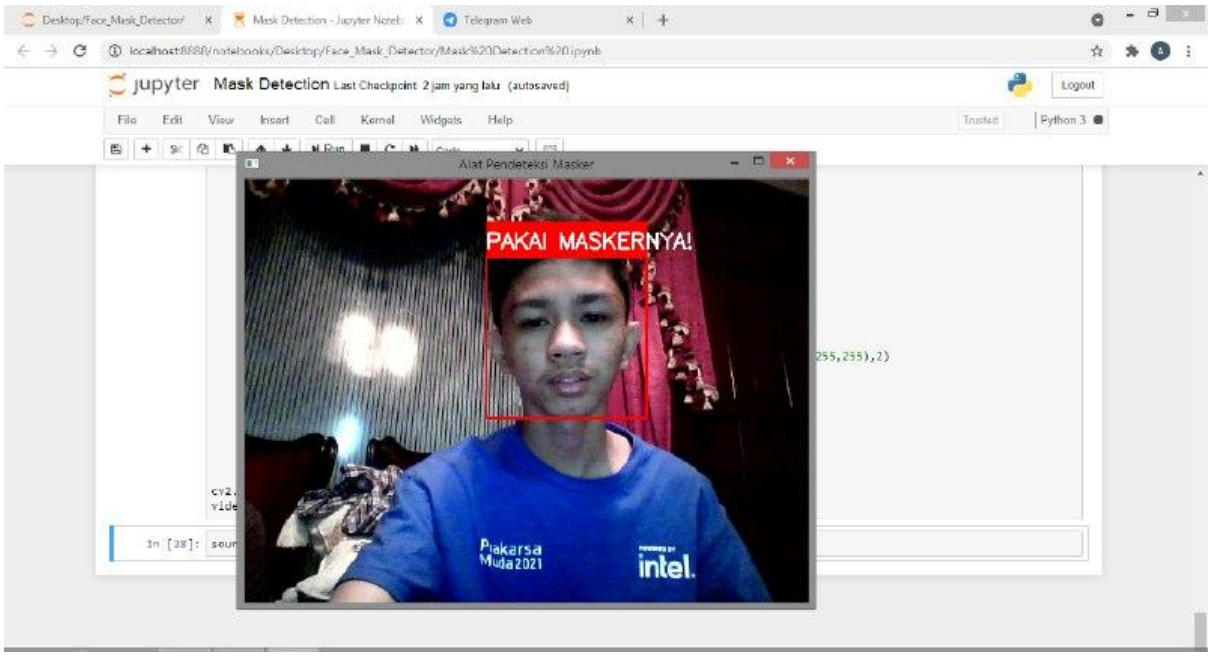
Load the model, cascade classifier for face detection, dictionary to show the results

```
In [14]: video_capture = cv2.VideoCapture(0)

while(True):
    ret, img = video_capture.read()
    gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces=face_clsfr.detectMultiScale(gray,1.3,5)

    for (x,y,w,h) in faces:

        face_img=gray[y:y+w,x:x+w]
        resized=cv2.resize(face_img,(100,100))
        normalized=resized/255.0
        reshaped=np.reshape(normalized,(1,100,100,1))
        result=model.predict(reshaped)

        label=np.argmax(result,axis=1)[0]

        cv2.rectangle(img,(x,y),(x+w,y+h),color_dict[label],2)
        cv2.rectangle(img,(x,y-40),(x+w,y),color_dict[label],-1)
        cv2.putText(img, labels_dict[label], (x, y-10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)


    cv2.imshow('Alat Pendeteksi Masker',img)
    key=cv2.waitKey(1)

    if(key==27):
        break

cv2.destroyAllWindows()
video_capture.release()
```

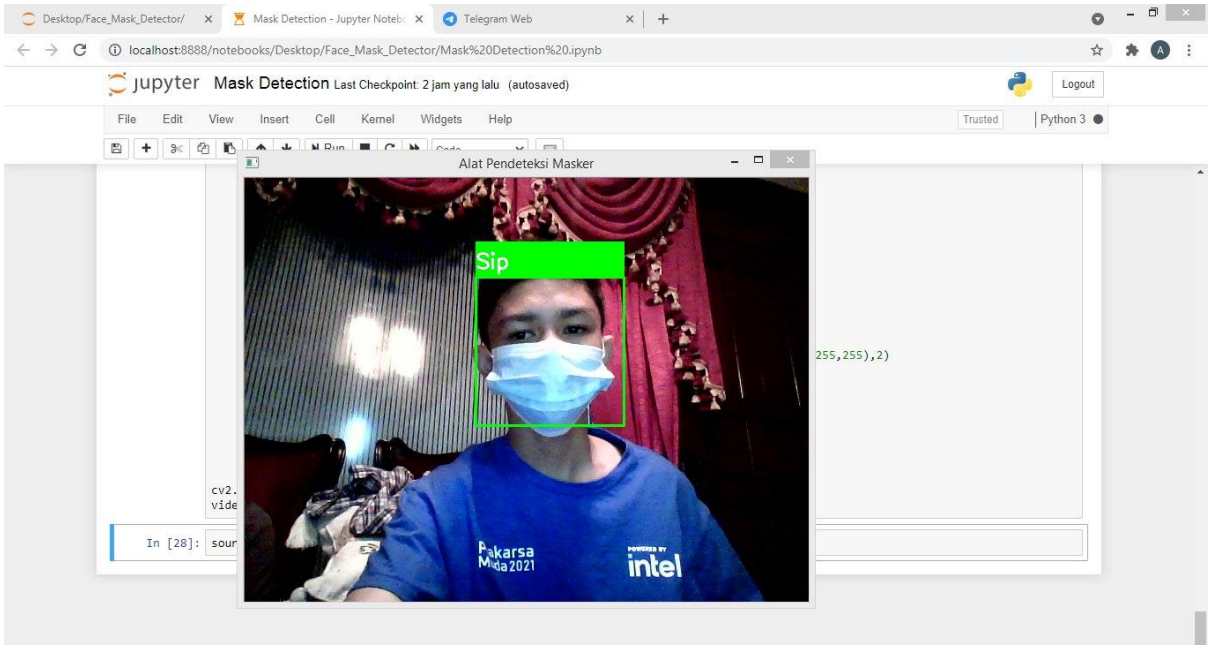
Final step is starting the video capture

4. Results

The results and findings would be presented here.
The CNN algorithm was carried out with the 2 classifications and we got the following results:



The capture with non-wear mask



The capture with wear mask

The results are quite similar to our intuitions. The first classification contains the captured image of people wearing a mask. The second classification contains the captured image of people not wearing a mask. So, we have successfully divided people wearing masks and non-wear masks.

5. Conclusion

Covid-19, pandemic worldwide can be checked easily by controlling our physical distancing. The project focused on controlling self safety (tracing people wear masks) by human visual recognition with CNN algorithm, a simple and useful classification algorithm, has been used here to classify human visual recognition into people wearing masks and non-wear masks so that people can keep track of when they are violating health protocols.

6. Bibliography

1. Unicef web blog. (2020, March). https://www.unicef.org/indonesia/id/laporan/panduan-sekolah-untuk-pencegahan-coronavirus
2. Google.com, Covid-19 Prevention https://www.google.com/search?q=covid+19+prevention&oq=covid&aqs=chrome.1.69i57j69i59l2j0i131i433i512l2j69i60l3.2794j0j7&sourceid=chrome&ie=UTF-8
3. Medium.com, Rishi Prasana, Face Mask Detection using OpenCV and Keras (2020, May 26) https://medium.com/@riship2009/face-mask-detection-using-opencv-and-keras-11ea8f565677
4. YouTube, nicholas renotte, Build a Face Detector in 20 Minutes with Watson Studio (2020, August 13) https://www.youtube.com/watch?v=T9KfYaS9hwQ