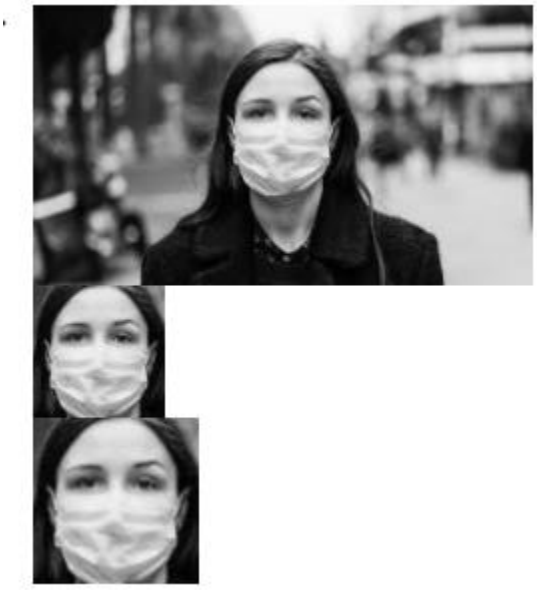
Siddharth Khachane

Mask Detection CNN

Simple Implementation (Raw Grayscale Images as the Input)

Problem→

These were a lot of useless features which were taken into account like the exact pixels of face.

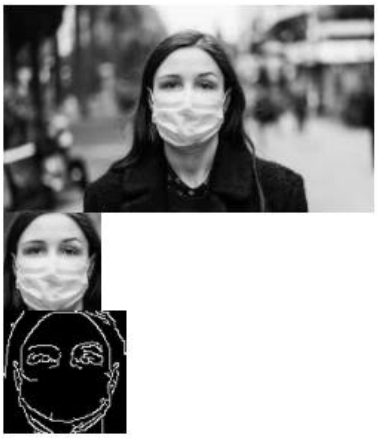


Solution→

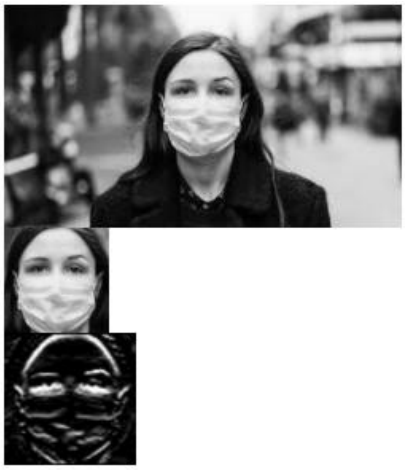
We could use Edge Detection to have only the essential features in the image.

Different Edge detections

Canny→



Sobel→



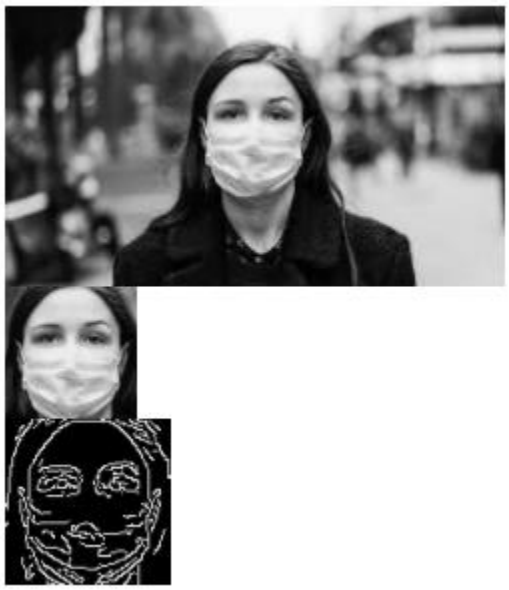
Solution→

I tried many edge detection techniques and so far canny detected the edges properly

Canny Threshold problem→

Low difference between lower and upper bound→

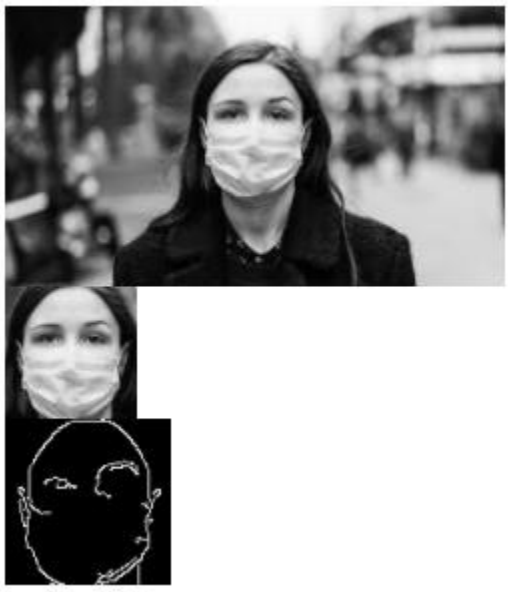
Cv2.canny(img,100,150)



Masks have folds on them so if the difference between lower and upper bounds of the thresholds is low, then these folds might be detected as edges and thus result in the wrong prediction as the model might confuse these edges as facial features.

High difference between lower and upper bound→

Cv2.canny(img,100,255)



If the difference between the threshold bounds is very high then it might result in the loss of essential features.

Problem→

For Each Image, these threshold limits vary for getting the optimal edges. We cannot set a common Threshold limit for all images

Solution→

We should use some parameters which find the correct value for the threshold limits.

Using median to compute the thresholds

```
def auto_canny(image, sigma=0.33):
  # compute the median of the single channel pixel intensities
  v = np.median(image)
  # apply automatic Canny edge detection using the computed median
  lower = int(max(0, (1.0 - sigma) * v))
  upper = int(min(255, (1.0 + sigma) * v))
  edged = cv2.Canny(image, lower, upper)
  # return the edged image
  return edged
```

sigma is fixed as 0.33 as it tends to give good results on mask detection images.

The amount by which we want the effect of the median depends upon sigma

Max(0,0.67*median) gives the lower threshold.

Min(255,0.67*median) gives the higher threshold.

[[1. 0.]]

[[0. 1.]]

[[1. 0.]]

[[1. 0.]]

[[1. 0.]]

[[1. 0.]]