



# **iJRASET**

International Journal For Research in  
Applied Science and Engineering Technology



---

# **INTERNATIONAL JOURNAL FOR RESEARCH**

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

---

**Volume: 10    Issue: 1    Month of publication: January 2022**

**DOI: <https://doi.org/10.22214/ijraset.2022.39880>**

**[www.ijraset.com](http://www.ijraset.com)**

**Call:  08813907089**

**E-mail ID: [ijraset@gmail.com](mailto:ijraset@gmail.com)**

# Face Mask Detection Using Deep Learning and Computer Vision

Prof. Roshan R. Kolte<sup>1</sup>, Mr. Harshal P. Kadu<sup>4</sup>, Mrs. Vaishali V. Akhare<sup>2</sup>, Mr. Pushkar N. Dhande<sup>3</sup>, Mr. Sagar Rokde<sup>5</sup>  
<sup>1, 2, 3, 4, 5</sup>Information Technology, KDKCE, Nagpur, India

**Abstract:** COVID-19 pandemic has rapidly affected our day-to-day life the world trade and movements. Wearing a face mask is very essentials for protecting against virus. People also wear mask to cover themselves in order to reduce the spread of covid virus. The corona virus covid-19 pandemic is causing a global health crisis so the effective protection method is wearing a face mask in public area according to the world health organization (WHO). The covid-19 pandemic forced government across the world to impose lockdowns to prevent virus transmission report indicates that wearing face mask while at work clearly reduce the risk of transmission .we will use the dataset to build a covid-19 face mask detector with computer vision using python,opencv,tensorflow,keras library and deep learning. Our goal is to identify whether the person on image or live video stream is wearing mask or not wearing face mask this can help to society and whole organization to avoid the transfer of virus one person to antother.we used computer vision and deep learning modules to detect a with mask image and without mask image.

**Keywords:** face detection, face recognition, CNN, SVM, opencv, python, tensorflow, keras.

## I. INTRODUCTION

The trend of wearing face masks in public is rising due to covid-19 corona virus epidemic all over the world. Before covid-19 people used wear a mask to protect their faces from pollution and sun rays. Some people are self-conscious about their look they hides their faces. More than five million case were affected by covid-19 in less than 6 month across 188 countries the virus spread through close contact and crowded and overcrowded area like public place, mall, college etc. People are forced by laws to wear face mask in public in many countries these rules and laws were developed as an action to the exponential growth in case and death in many area. Here we introduce a face mask detection model that is based on computer vision and deep learning the proposed mode can be integrated with surveillances camera to impede to covid-19 transmission by allowing the detection of people who are wearing mask or not wearing a face masks. We used deep learning and computer vision model with opencv tensor flow and keras we used dataset for training and live stream video detect people are wearing a face mask or not our project have high accuracy we used less time for execution.

## II. TECHNOLOGY USED

- 1) OpenCV
- 2) Keras
- 3) Python
- 4) VS code
- 5) Dataset
- 6) Webcam

Algorithm like svm, cnn, face detector and face recognition.

## III. MAIN MODULE OF SYSTEM

- A. Creating a dataset
  - B. Train image from dataset
  - C. Train face mask detector
  - D. Apply face mask detector
  - E. Detect face in live video stream
  - F. Detect people wear a mask or they have without mask.
- **Purpose:** The main purpose of this system to detect the face of people who wear a face mask or not in public place as well as overcrowded place. If they are not were mask then take action against them who not wear a mask this can help as to reduce the transmission of virus and contact of people are avoid it help as to building a new healthy society.

#### IV. METHODOLOGY AND OVERVIEW OF SYSTEM

The various step are performed in the methodology of project.

- 1) We have create a dataset with mask and without mask
- 2) Train this dataset with accuracy.
- 3) Apply face detector recogniser algorithm to dataset.
- 4) Apply face mask detector
- 5) Train mask detector detect the face in webcam with mask and without mask from dataset as well as live video stream.

The various image with high accuracy are created with different posture and store in database and name that image as training dataset with mask and without mask. They are store in database or in folder.

Train the image with svm algorithm and cnn algorithm train it with mask dataset and without mask dataset. Face detector is used to train the image from dataset.

After it recognised the face and apply face detector algorithm detection and training of dataset are done and show the people who wear a mask and not wearing face mask.

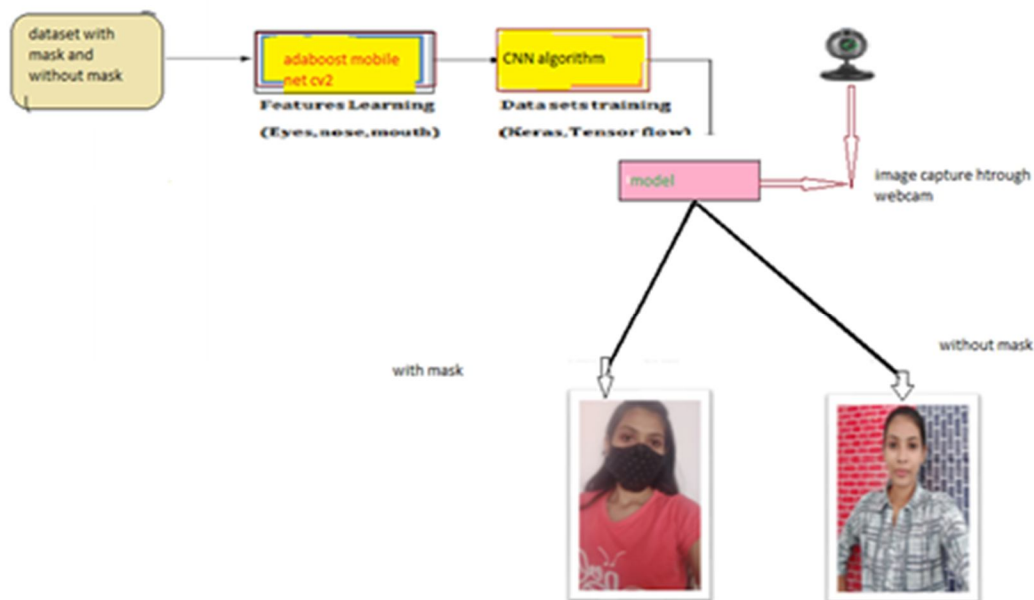
Face detector apply to all image then select one and train it they show that the people wearing a face mask show green rectangular box and those who are not wearing a mask they show red rectangular box around their face.

We also detect face in live stream people wearing mask show green rectangle and people without mask show red rectangle and show warning message to wear their mask.

#### V. ALGORITHM USED

We used face mask detection and face recognition and support vector machine (SVM) and CNN convolutional neural network algorithm for mask detection by using python opencv they identify the image with high accuracy and increase the performance of project cnn capture the image in webcam and recognised face detect face mask and tensorflow, keras, opencv are used in computer vision and deep learning.

- 1) CNN algorithm CNN architecture capable of detecting masked and unmasked faces and can be integrated with pre-installed CCTV cameras.
- 2) Ad boost algorithm and face detection algorithm are used.
- 3) Ad boost was invented by Freund and Schapire in 1997.
- 4) Ad boost was applied to face detection with modification by viola and jones in 2001.
- 5) The detection speed of 0.07 second per frame of size ~300\*300 on a standards desktop in 2001 to 2004.



## VI. EXECUTION PROCESS

The given dataset are train with the face mask detector in webcam as well as live video stream.

The first dataset are with mask dataset and select the one of them and then detect the person face show image are with mask or without mask. And if the mask is not wear they show a warning with red rectangular box around the face.



Picture from dataset with mask and without mask they are train with face mask detector with webcam and then shows a result as green rectangle box with mask image and around without mask image and give the warning to wear a mask.

### A. Execution

1) The data is collected with masks and without masks.

2) The software required:

Anaconda navigator, Google coolab, and python library like keras, opencv, Tensorflow, Numpy, matplotlib.

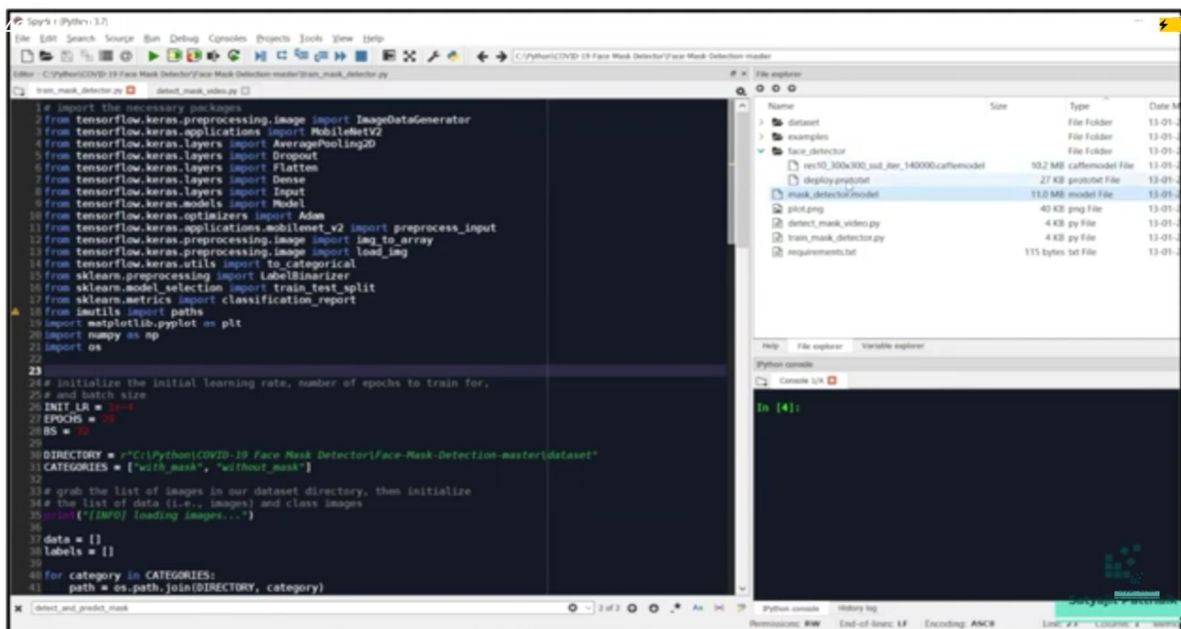
The major requirement for implementing this project using python programming language and deep learning, computer vision.cnn algorithm is used for implementation.

a) *Implementation:* Dataset collecting: - we collect no. of data sets with face mask and without masks. We get high accuracy depending on no. Of image.

b) *Datasets Extracting:* We can extract features using mobile net v2 of mask and no mask sets.

c) *Models Training:* We will train the model using opencv, keras (python library).

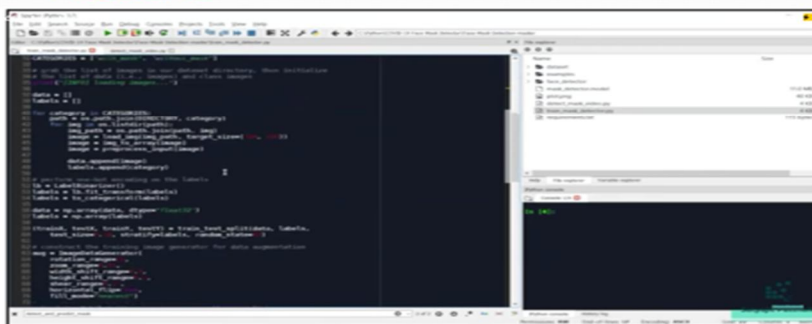
d) *Facemask Detection:* We can detect Pre-processing image and also detect live video. If people wearing masks they permit otherwise they take an action in this way we prevent transmission of virus.



```

1 # Import the necessary packages
2 from tensorflow.keras.preprocessing.image import ImageDataGenerator
3 from tensorflow.keras.applications import MobileNetV2
4 from tensorflow.keras.layers import AveragePooling2D
5 from tensorflow.keras.layers import Dropout
6 from tensorflow.keras.layers import Flatten
7 from tensorflow.keras.layers import Dense
8 from tensorflow.keras.layers import Input
9 from tensorflow.keras.models import Model
10 from tensorflow.keras.optimizers import Adam
11 from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
12 from tensorflow.keras.preprocessing.image import img_to_array
13 from tensorflow.keras.preprocessing.image import load_img
14 from tensorflow.keras.utils import to_categorical
15 from sklearn.preprocessing import LabelBinarizer
16 from sklearn.model_selection import train_test_split
17 from sklearn.metrics import classification_report
18 from osutils import paths
19 import matplotlib.pyplot as plt
20 import numpy as np
21 import os
22
23
24 # Initialize the initial learning rate, number of epochs to train for,
25 # and batch size
26 INIT_LR = 0.001
27 EPOCHS = 10
28 BS = 32
29
30 DIRECTORY = r"C:\Python\COVID-19 Face Mask Detector\Face-Mask-Detection-master\dataset"
31 CATEGORIES = ["with_mask", "without_mask"]
32
33 # Grab the list of images in our dataset directory, then initialize
34 # the list of data (i.e., images) and class images
35 print("[INFO] loading images...")
36
37 data = []
38 labels = []
39
40 for category in CATEGORIES:
41     path = os.path.join(DIRECTORY, category)
  
```

## Result



```
function [net, input_names, output_names] = create_network(layers)
% Create a neural network with the specified layers.
% layers is a cell array of strings, where each string is the name of a layer.
% The first layer must be 'input' and the last layer must be 'output'.
% The network is created using the 'trainnet' function.

% Create the input layer.
input_layer = layer('input', layers{1});

% Create the hidden layers.
for i = 2:length(layers)
    hidden_layer = layer('fully_connected', layers{i});
    net = layerConnection(net, input_layer, hidden_layer);
end

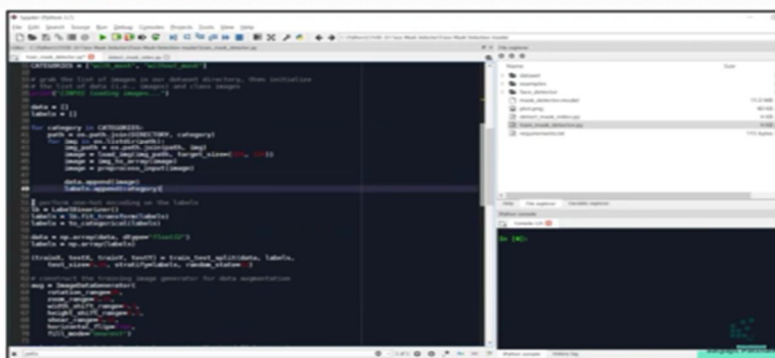
% Create the output layer.
output_layer = layer('fully_connected', layers{length(layers)});
net = layerConnection(net, hidden_layer, output_layer);

% Create the training options.
options = optimoptions('nnet_train', 'max_epochs', 1000, 'display', 'none');

% Train the network.
net = trainnet(net, input_names, output_names, options);

% Return the network and the names of the layers.
return [net, input_names, output_names];
```

## Result



```
function [net, input_names, output_names] = create_network(layers)
% Create a neural network with the specified layers.
% layers is a cell array of strings, where each string is the name of a layer.
% The first layer must be 'input' and the last layer must be 'output'.
% The network is created using the 'trainnet' function.

% Create the input layer.
input_layer = layer('input', layers{1});

% Create the hidden layers.
for i = 2:length(layers)
    hidden_layer = layer('fully_connected', layers{i});
    net = layerConnection(net, input_layer, hidden_layer);
end

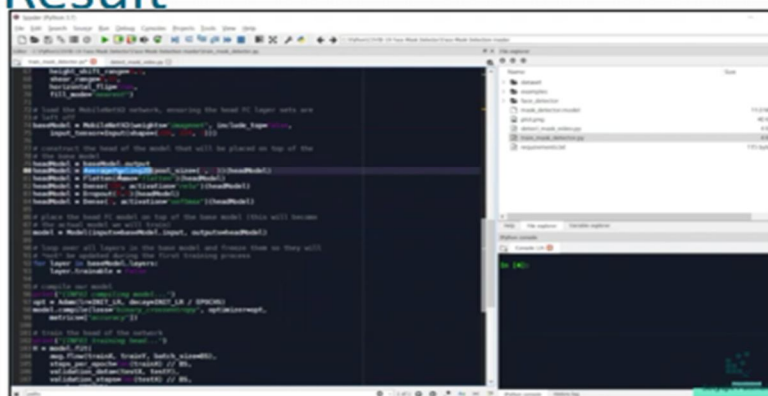
% Create the output layer.
output_layer = layer('fully_connected', layers{length(layers)});
net = layerConnection(net, hidden_layer, output_layer);

% Create the training options.
options = optimoptions('nnet_train', 'max_epochs', 1000, 'display', 'none');

% Train the network.
net = trainnet(net, input_names, output_names, options);

% Return the network and the names of the layers.
return [net, input_names, output_names];
```

## Result



```
function [net, input_names, output_names] = create_network(layers)
% Create a neural network with the specified layers.
% layers is a cell array of strings, where each string is the name of a layer.
% The first layer must be 'input' and the last layer must be 'output'.
% The network is created using the 'trainnet' function.

% Create the input layer.
input_layer = layer('input', layers{1});

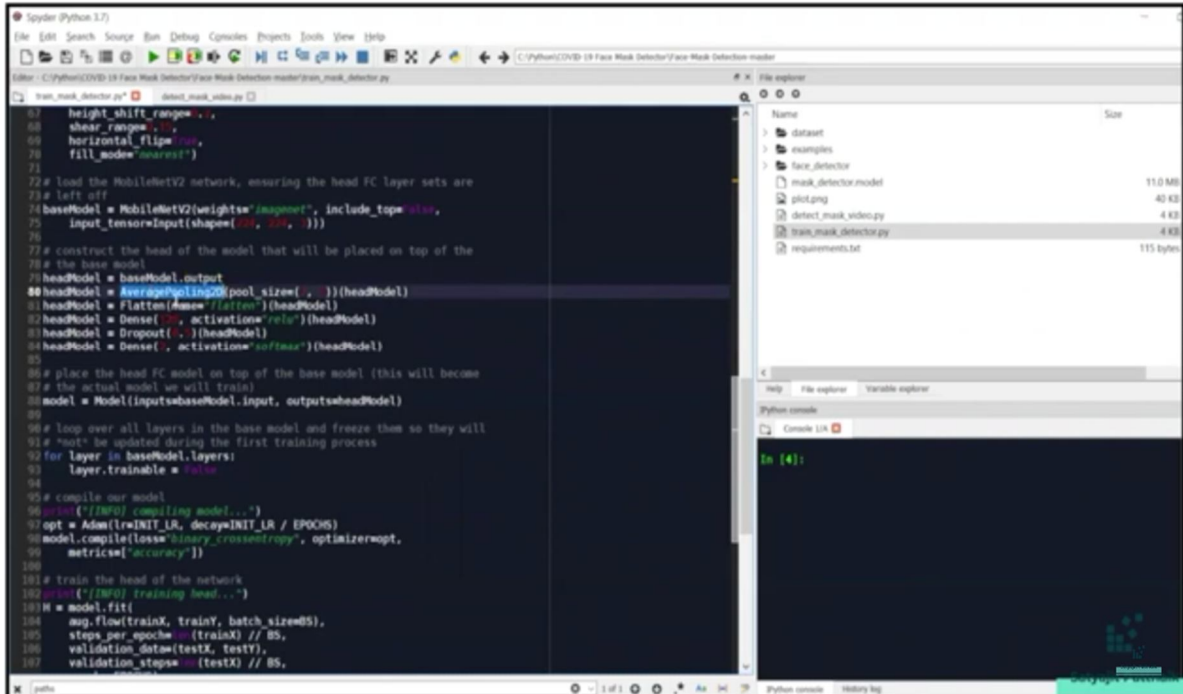
% Create the hidden layers.
for i = 2:length(layers)
    hidden_layer = layer('fully_connected', layers{i});
    net = layerConnection(net, input_layer, hidden_layer);
end

% Create the output layer.
output_layer = layer('fully_connected', layers{length(layers)});
net = layerConnection(net, hidden_layer, output_layer);

% Create the training options.
options = optimoptions('nnet_train', 'max_epochs', 1000, 'display', 'none');

% Train the network.
net = trainnet(net, input_names, output_names, options);

% Return the network and the names of the layers.
return [net, input_names, output_names];
```

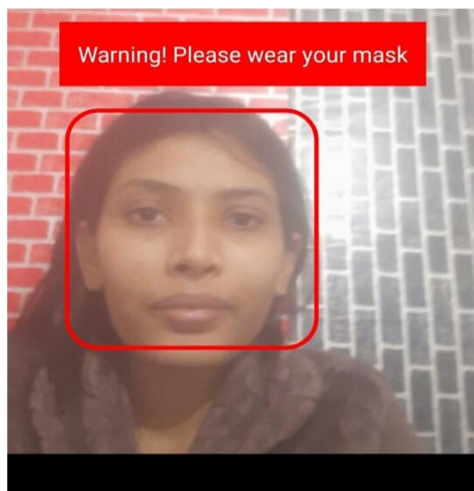


```

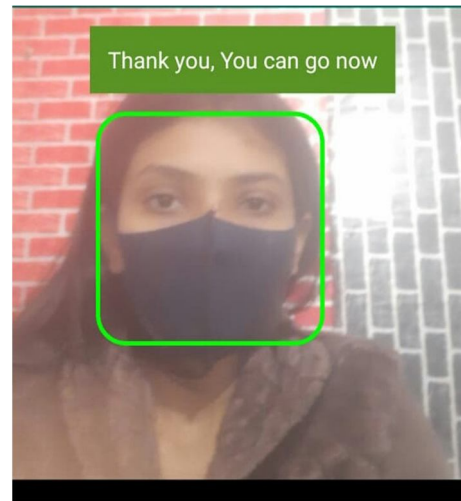
1 train_mask_detector.py
2 detect_mask_video.py
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

**VII. OUTPUT**



Without mask



with mask

**VIII. BENEFITS**

- 1) Manual monitoring is very difficult for officers to check whether the people are wearing mask or not .so in our technique we used live camera to detect people without mask easily and also prevent from virus transmission.
  - 2) It has fast and high accuracy
  - 3) This system can be implemented in public places.
- We can keep people safe from our technique.

**IX. CONCLUSION**

- 1) By the development of face mask detection we can detect if the person is wearing a face mask and allow their entry would be of great help to the society.
- 2) We can prevent peoples from virus transmission through this system.

## REFERENCES

- [1] Ahmed I., Ahmad M., Rodrigues J.J., Jeon G., Din S. A deep learning-based social distance monitoring framework for COVID-19. Sustainable Cities and Society. 2020 [[PMC free article](#)] [[PubMed](#)] [[Google Scholar](#)]
- [2] Alom M.Z., Taha T.M., Yakopcic C., Westberg S., Sidike P., Nasrin M.S....Asari V.K. 2018. The history began from alexnet: A comprehensive survey on deep learning approaches. ArXiv preprint arXiv: 1803.01164. [[Google Scholar](#)]
- [3] Anisimov D., Khanova T. Towards lightweight convolutional neural networks for object detection. 2017 14th IEEE international conference on advanced video and signal based surveillance (AVSS); IEEE; 2017. pp. 1–8. August. [[Google Scholar](#)]
- [4] Chen D., Ran S., Wei Y., Cao X., Sun J. European conference on computer vision. Springer; Cham: 2014. Joint cascade face detection and alignment; pp. 109–122. September. [[Google Scholar](#)]
- [5] Chen D., Hua G., Wen F., Sun J. European conference on computer vision. Springer; Cham: 2016. Supervised transformer network for efficient face detection; pp. 122–138. October. [[Google Scholar](#)]
- [6] Ge S., Li J., Ye Q., Luo Z. Detecting masked faces in the wild with lle-cnns. Proceedings of the IEEE conference on computer vision and pattern recognition. 2017:2682–2690. [[Google Scholar](#)]
- [7] Ge X.Y., Pu Y., Liao C.H., Huang W.F., Zeng Q., Zhou H....Chen H.L. Evaluation of the exposure risk of SARS-CoV-2 in different hospital environment. Sustainable Cities and Society. 2020;61 [[PMC free article](#)] [[PubMed](#)] [[Google Scholar](#)]
- [8] Ghiasi G., Fowlkes C.C. Occlusion coherence: Localizing occluded faces with a hierarchical deformable part model. Proceedings of the IEEE conference on computer vision and pattern recognition. 2014:2385–2392. [[Google Scholar](#)]
- [9] Gulcehre C., Moczulski M., Denil M., Bengio Y. International conference on machine learning, 2016. Noisy activation functions; pp. 3059–3068. June. [[Google Scholar](#)]
- [10] Ahmed I., Ahmad M., Rodrigues J.J., Jeon G., Din S. A deep learning-based social distance monitoring framework for COVID-19. Sustainable Cities and Society. 2020 [[PMC free article](#)] [[PubMed](#)] [[Google Scholar](#)]



10.22214/IJRASET



45.98



IMPACT FACTOR:  
7.129



IMPACT FACTOR:  
7.429



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24\*7 Support on Whatsapp)