Mask Detection and Categorization during the COVID-19 Pandemic Using Deep Convolutional Neural Network

Detección y categorización de máscaras durante la pandemia del COVID-19 utilizando una red neuronal convolucional profunda

Kamil Dimililer1,3, and Devrim Kayali2

ABSTRACT
With COVID-19 spreading all over the world and restricting our daily lives, the use of face masks has become very important, as it is an efficient way of slowing down the spread of the virus and an important piece to continue our daily tasks until vaccination is completed. People have been fighting this disease for a long time, and they are bored with the precautions, so they act carelessly. In this case, automatic detection systems are very important to keep the situation under control. In this research, deep learning models are trained with as little input data as possible in order to obtain an accurate face mask-wearing condition classification. These classes are mask-correct, mask wrong, and no mask, which refers to proper face mask use, improper face mask use, and no mask use, respectively. DenseNets, EfficientNets, InceptionResNetV2, InceptionV3, MobileNets, NasNets, ResNets, VGG16, VGG19, and Xception are the networks used in this study. The highest accuracy was obtained by the InceptionResNetV2 and Xception networks, with 99.6%. When other performance parameters are taken into consideration, the Xception network is a step forward. VGG16 and VGG19 also show an accuracy rate over 99%, with 99.1 and 99.4%, respectively. These two networks also had higher FPS and the two lowest initialization times during implementation. A comparison with recent studies was also carried out to evaluate the obtained accuracy. It was found that a higher accuracy can be obtained with the possible minimum input size.

Keywords: COVID-19, mask detection, deep learning, classification

RESUMEN
Con el COVID-19 extendiéndose por todo el mundo y restringiendo nuestra vida diaria, el uso de mascarillas se ha vuelto muy importante, pues es una forma eficiente de frenar la propagación del virus, y una pieza importante para continuar con nuestras tareas diarias hasta que se complete la vacunación. La gente ha estado luchando contra la enfermedad durante mucho tiempo y se aburre con las precauciones, por lo que actúa con descuido. En este caso, los sistemas de detección automática son muy importantes para mantener la situación bajo control. En esta investigación se entrenan modelos de aprendizaje profundo con el mínimo de datos de entrada posibles para obtener una clasificación precisa de las condiciones de uso de las mascarillas. Estas clases son maskcorrect, maskwrong y nomask, que se refieren al uso adecuado, a un uso inadecuado y al no uso de la mascarilla facial, respectivamente. DenseNets, EfficientNets, InceptionResNetV2, InceptionV3, MobileNets, NasNets, ResNets, VGG16, VGG19 y Xception son las redes utilizadas en este estudio. La mayor precisión la obtuvieron las redes InceptionResNetV2 y Xception, con un 99,6 %. Cuando se tienen en cuenta otros parámetros de rendimiento, la red Xception un paso adelante. VGG16 y VGG19 presentan una tasa de precisión superior al 99 %, con 99.1 y 99.4 % respectivamente. Estas dos redes también presentaron FPS más altos y los dos tiempos de inicialización más bajos en la implementación. También se realizó una comparación con estudios recientes para evaluar la precisión obtenida. Se encontró que se puede obtener una mayor precisión con el mínimo tamaño de entrada posible.

Palabras clave: COVID-19, detección de mascarillas, aprendizaje profundo, clasificación

Received: March 25th 2022
Accepted: October 31st 2022

Introduction
Our lifestyles started changing with the advent of COVID-19, a very dangerous virus that quickly spread all over the world. Due to its high infection rate and severity, precautions were taken at a global level, and restrictions were put into effect. The use of face masks was one of the most important and effective ways for slowing down the infection from the very beginning of the pandemic (Cheng et al., 2020).

Despite the fact that vaccination continues all over the world, the use of masks is still important, since sufficient vaccination has not yet been reached and the virus can mutate and change. As we are trying to return to our daily lives while coexisting with the disease, we still have to take some precautions to avoid any rapid spread. This means that serious control measures are needed, especially in public places. Automated control is key to obtain fast, reliable, and continuously working systems. Artificial intelligence (AI) makes it possible to obtain such systems in many different areas. AI-based image processing has been applied in various cases (Dimililer, 2022; Dimililer and Kayali, 2021;
Kodali and Dhanekula (2021) proposed a deep learning-based 
method for crowd counting and face mask detection.

Shete and Milosavljvec (2020) also studied a system that 
detects both face masks and social distancing, using DSF,
and Normalizes it. Then, the proposed CNN classifies the 
face images with and without face masks. The validation 
accuracy of the proposed model is 96%.

Pinki and Garg (2020) developed a deep-learning model for 
a face mask detection system by fine-tuning MobileNetV2. 
The developed model can be integrated into systems in 
public areas to identify whether the people are wearing 
masks. If a person is not wearing a mask, a notification 
is sent to the relevant authorities.

Sen and Patidar (2020) developed a deep learning-based 
method using MobileNetV2 and building four fully 
connected layers on top of it. The developed model detects 
people with or without face masks from images and video 
streams with an accuracy of 79.24%.

Yadav (2020) proposed a system that uses both deep 
learning and geometric techniques for detection, tracking, 
and validation purposes. The system performs real-time 
monitoring to detect face masks and social distancing 
between people.

Shete and Milosavljvec (2020) also studied a system that 
detects both face masks and social distancing, using DSF, 
YOLOv3, and ResNet classifier pre-trained models. This 
system provides the number of non-violating and violating 
people as an output with a confidence score of 100%.

Srinivasan et al. (2021) proposed a system that uses 
Dual Shot Face Detector (DSFD), Density-based spatial 
clustering of applications with noise (DBSCAN), YOLOv3, 
and MobileNetV2 for an effective solution regarding face 
real-time application in terms of both accuracy and average 
interference time.

Mohan et al. (2021) proposed a tiny CNN architecture that 
can be used on devices with constrained resources. The 
proposed architecture was compared to the SqueezeNet 
and the Modified SqueezeNet (a smaller version of the 
former). According to the results, the proposed method 
outperformed the SqueezeNet and the Modified SqueezeNet 
in both size and accuracy.

Amin et al. proposed a real-time deep learning-based 
method for crowd counting and face mask detection.

Snyder and Husari (2021) developed an automated detection 
model and implemented it on a mobile robot named 
Thor. The developed method consists of three components:
the detection of human existence, the detection and 
evaluation of the human face, and the classification of 
masked/unmasked faces. In the first component, ResNet50 
is used with Feature Pyramid Network (FPN). The second 
component detects and extracts human faces with multi-
task convolutional neural networks (MT-CNNs). The last 
component consists of a trained CNN classifier for the 
detection phase.

Khamlae et al. developed and trained a CNN with a dataset 
consisting of 848 images with a 416x416 resolution. An 
accuracy rate of 81% was obtained with the trained model, 
which was then implemented at the front gate of a campus 
building.

Kodal and Dhanekula (2021) proposed a deep learning-
based network for face mask detection. This method 
consists of two parts. The first part, the pre-processing 
step, converts the RGB image to a grayscale one and resizes 
and Normalizes it. Then, the proposed CNN classifies the 
face images with and without face masks. The validation 
accuracy of the proposed model is 96%.

Pinki and Garg (2020) developed a deep-learning model for 
a face mask detection system by fine-tuning MobileNetV2. 
The developed model can be integrated into systems in 
public areas to identify whether the people are wearing 
masks. If a person is not wearing a mask, a notification 
is sent to the relevant authorities.

Sen and Patidar (2020) developed a deep learning-based 
method using MobileNetV2 and building four fully 
connected layers on top of it. The developed model detects 
people with or without face masks from images and video 
streams with an accuracy of 79.24%.

Yadav (2020) proposed a system that uses both deep 
learning and geometric techniques for detection, tracking, 
and validation purposes. The system performs real-time 
monitoring to detect face masks and social distancing 
between people.

Shete and Milosavljvec (2020) also studied a system that 
detects both face masks and social distancing, using DSF, 
YOLOv3, and ResNet classifier pre-trained models. This 
system provides the number of non-violating and violating 
people as an output with a confidence score of 100%.

Srinivasan et al. (2021) proposed a system that uses 
Dual Shot Face Detector (DSFD), Density-based spatial 
clustering of applications with noise (DBSCAN), YOLOv3, 
and MobileNetV2 for an effective solution regarding face 
real-time application in terms of both accuracy and average 
interference time.

Mohan et al. (2021) proposed a tiny CNN architecture that 
can be used on devices with constrained resources. The 
proposed architecture was compared to the SqueezeNet 
and the Modified SqueezeNet (a smaller version of the 
former). According to the results, the proposed method 
outperformed the SqueezeNet and the Modified SqueezeNet 
in both size and accuracy.

Amin et al. proposed a real-time deep learning-based 
method for crowd counting and face mask detection.

Snyder and Husari (2021) developed an automated detection 
model and implemented it on a mobile robot named 
Thor. The developed method consists of three components:
the detection of human existence, the detection and 
evaluation of the human face, and the classification of 
masked/unmasked faces. In the first component, ResNet50 
is used with Feature Pyramid Network (FPN). The second 
component detects and extracts human faces with multi-
task convolutional neural networks (MT-CNNs). The last 
component consists of a trained CNN classifier for the 
detection phase.

Khamlae et al. developed and trained a CNN with a dataset 
consisting of 848 images with a 416x416 resolution. An 
accuracy rate of 81% was obtained with the trained model, 
which was then implemented at the front gate of a campus 
building.

Kodal and Dhanekula (2021) proposed a deep learning-
based network for face mask detection. This method 
consists of two parts. The first part, the pre-processing 
step, converts the RGB image to a grayscale one and resizes 
and Normalizes it. Then, the proposed CNN classifies the 
face images with and without face masks. The validation 
accuracy of the proposed model is 96%.

Pinki and Garg (2020) developed a deep-learning model for 
a face mask detection system by fine-tuning MobileNetV2. 
The developed model can be integrated into systems in 
public areas to identify whether the people are wearing 
masks. If a person is not wearing a mask, a notification 
is sent to the relevant authorities.

Sen and Patidar (2020) developed a deep learning-based 
method using MobileNetV2 and building four fully 
connected layers on top of it. The developed model detects 
people with or without face masks from images and video 
streams with an accuracy of 79.24%.

Yadav (2020) proposed a system that uses both deep 
learning and geometric techniques for detection, tracking, 
and validation purposes. The system performs real-time 
monitoring to detect face masks and social distancing 
between people.

Shete and Milosavljvec (2020) also studied a system that 
detects both face masks and social distancing, using DSF, 
YOLOv3, and ResNet classifier pre-trained models. This 
system provides the number of non-violating and violating 
people as an output with a confidence score of 100%. 

Srinivasan et al. (2021) proposed a system that uses 
Dual Shot Face Detector (DSFD), Density-based spatial 
clustering of applications with noise (DBSCAN), YOLOv3, 
and MobileNetV2 for an effective solution regarding face
mask and social distancing detection. According to an evaluation of its performance, the system obtained accuracy and F1 scores of 91.2 and 90.79%, respectively. 

Meivel et al. (2021) proposed a real-time social distancing measurement and face mask detection system using Matlab, RCNN, Fast RCNN, and Faster RCNN algorithms were chosen to this effect.

Wang et al. (2021) proposed a face mask detection solution dubbed Web-based efficient AI recognition of masks (WearMask). It is an in-browser, serverless, edge-computing-based application that requires no software installation and can be used by any device with an internet connection and access to web browsers. This framework integrates deep learning (YOLO), a high-performance neural network inference computing framework (NCNN), and a stack-based virtual machine (WebAssembly).

Dey et al. (2021) proposed a multi-phase deep learning-based model for face mask detection in images and video streams, which was named MobileNet Mask. According to their experimental results, MobileNet Mask achieved about 93% accuracy with 770 validation samples, and about 100% accuracy with 276 validation samples.

Naual et al. (2021) conducted a comparative study on face mask detection using support vector machines (SVM), k-nearest neighbors (KNN), and deep CNNs (DCNN). Although CNN required a longer execution time compared to KNN and SVM, it reported the best average performance, with an accuracy of 96.83%.

Vijitkunswat and Chantngarm (2020) also conducted a comparative study on the performance of SVM, KNN, and MobileNet for real-time face mask detection. According to the results, MobileNet has the best accuracy with regard to both images and real-time video inputs.

Singh et al. (2021) studied and implemented two different real-time face mask detection models. One of them involved YOLOv4, and the second one used MobileNetV2 and a CNN. A comparison of the experimental results showed that MobileNetV2 and CNN yielded better results, with an accuracy of 98%, in comparison with the accuracy of YOLOv4 (88.92%).

Sivakumar et al. made a deep learning based study on face mask detection by using Viola-Jones algorithm and the Convolution Neural Networks (CNN).

Nagrath et al. (2021) proposed SSDMNV2, an approach based on OpenCV and deep learning. In this method, a Single Shot Multibox Detector is used as a detector, and the MobileNetV2 network is used as a classifier. The accuracy and F1 score of the proposed system are 92.64 and 93%, respectively.

Sanjaya and Rakhmawan (2020) also used MobileNetV2 to develop a face mask detection algorithm in order to fight COVID-19. Their model can discriminate people with and without a face mask with an accuracy of 96.83%.

Sakshi et al. (2021) also used MobileNetV2 to implement a face mask detection model to be used both on static pictures and real-time videos. Venukeswarlu et al. (2020) used MobileNet and a global pooling block to develop a face mask detection model. The global pooling layer flattens the feature vector, and a fully connected dense layer and the softmax layer utilize classification. The system was evaluated with different datasets, and the obtained accuracies were 99 and 100%.

Rudraraju et al. (2020) proposed a face mask detection system to control various entrances of a facility using fog computing. Fog nodes are used for processing the video streams of the entrances. Haar-cascade-classifiers are used to detect faces in the frames, and each fog node consists of two MobileNet models. The first model classifies whether the person is wearing a mask, and if this model gives the output that the person is wearing a face mask, then the second model classifies whether the mask-wearing condition is proper. As a result of this two-level classifier, the person is allowed to enter the facility if they are properly wearing a mask. The accuracy of this system is about 90%.

Hussain et al. (2021) proposed a Smart Screening and Disinfection Walkthrough Gate (SSDWG), an IoT-based control and monitoring system. This system performs rapid screening, includes temperature measuring, and stores the record of suspected individuals. The screening system uses a deep learning model for face mask detection and the classification of people who are wearing face mask properly, improperly, or not wearing any mask. A transfer learning approach was used on CNN, ResNet-50, Inception v3, MobileNetV2, and VGG-16. Results showed that VGG-16 achieved the highest accuracy, with 99.81%, followed by the MobileNetV2 with 99.6%.

Adusumalli et al. (2021) studied a system that detects whether people are wearing face masks. The system is based on OpenCV and MobileNetV2. It also sends message to the person if they are not wearing a mask, and their face is stored in the database.

Das et al. (2020) used OpenCV and CNNs in their research to develop a face mask detection system. They used two different datasets, and the obtained accuracies were 95.77 and 94.58%.

Aydemir et al. (2022) used pre-trained DenseNet201 and ResNet101 for feature extraction. Then, they used an improved Relief selector to choose features in order to train an SVM classifier. They had three cases in their research: mask vs. no mask vs. improper mask; mask vs. no mask and improper mask; and mask vs. no mask. They obtained 95.95, 97.49, and 100% accuracy for these cases, respectively.

Wu et al. (2022) proposed an automatic facemask recognition and detection framework called FMD-Yolo. Im-Res2Net-101 was used to extract features, which is a modified Res2Net structure. Then, an aggregation network En-PAN was used to merge low-level details and high-level semantic information. According to their experimental results, at the intersection over union (IoU) level 0.5, FMD-Yolo had average precisions of 92 and 88.4% for two different datasets.

Mar-Cupido et al. (2022) conducted a study on classifying face masks used by people according to their type. They used the pre-trained models ResNet101v2, ResNet152v2, MobileNetv2, and NasNetMobile to classify the classes KN95, N95, cloth, surgical, and no mask. According to their results, ResNet101v2 and ResNet152v2 both had higher accuracies (98 and 97.37%, respectively) in comparison.
with NasNetMobile and MobileNetv2, both with 93.24% accuracy.

Agarwal et al. (2022) proposed a hybrid model with CNNs and SVMs. They used the medical mask dataset (MDD) and the real-world masked face recognition dataset (RMFD) for training and evaluating the proposed model. The obtained accuracy was 99.11%.

Crespo et al. (2022) carried out a study on both two- and three-class face mask-wearing conditions using deep learning while focusing on ResNets. The results showed that ResNet-18 had the best accuracies for both two and three classes, with 99.6 and 99.2% respectively.

Jayaswal and Dixit (2022) proposed a framework that uses a Single Shot Multibox Detector and Inception V3 (SSDIV3). They also proposed two versions of synthesized face mask datasets and compared SSDIV3 against VGG16, VGG19, Xception, and MobileNet. Their experimental results showed the higher accuracy of their proposed system, with 98%.

Singh et al. (2022) proposed a framework that uses cloud computing, fog computing, and deep neural networks. It is a smart and scalable system that can detect mask wearing and distancing violations. When the proposed framework was tested on video, it achieved a 98% accuracy.

Materials and methods

The goal of this study is to obtain an accurate face mask detection model that classifies three mask-wearing conditions with as little data as possible. This work selected models included in Keras, namely DenseNets, EfficientNets, InceptionResNetV2, InceptionV3, NasNets, MobileNets, ResNets, VGG16, VGG19, and Xception. These networks have different architectures, and some of them come in different versions that vary in parameters and depth. Moreover, these networks have differences in their default input shape, minimum input shape, and input data scale requirements. To train every network with minimal data, input shapes, and input data scale, requirements are examined and pre-processing is performed in order to obtain the necessary shapes.

Dataset

The dataset used in this research was created by the authors based on the Labeled Faces in the Wild (LFW) dataset (Huang et al., 2007). OpenCV was used for face and eyes detection in the LFW dataset. Once the faces were extracted from the images, face masks were added to the faces using eyes as reference points. To obtain the three classes needed, the first condition was to place the face mask properly covering the mouth and the nose of the person. In the second condition, the face mask was placed at a lower position, aiming to simulate improper face mask-wearing. For the last condition, the images were left untouched (no mask class).

Figure 1 shows examples of the classes in the dataset.

Pre-processing

The pre-processing stage was applied differently for each network depending on their properties. Each of the selected networks has a default input shape, and some of them also need the input data to be scaled between specific values for training. Since the goal was to train the network with as little data as possible, the minimum input shapes of each network were examined as shown in Table 1. The second most important thing was the rescaling of the input data in order to meet the network requirements. To this effect, each network’s input scale requirements were found (Table 2). Finally, the images were converted to grayscale in order to train with only one channel. Although these network inputs are generally three-channel images, grayscale images were used to reduce the input data. To summarize: first, the input image was converted to grayscale; then, the input image was resized to the minimum input size of the network; and the input data were rescaled according to the network’s input scale requirements. For networks with no input scale requirements, the input data were scaled between 0 and 1.

Network models

As mentioned at the beginning of this section, the networks selected for this study are DenseNets, EfficientNets, InceptionResNetV2, InceptionV3, MobileNets, NasNets, ResNets, VGG16, VGG19, and Xception.

Regarding the DenseNets, the models used in this research were DenseNet121, DenseNet169, and DenseNet201. The default input shape of these networks is (224, 224, 3), which means the expected input is a three-channel image of size 224x224. These networks can be trained with smaller images, but the size should not be smaller than 32x32. The EfficientNet models used in this research were EfficientNetB0, EfficientNetB1, EfficientNetB2, EfficientNetB3, EfficientNetB4, EfficientNetB5, EfficientNetB6, and EfficientNetB7. The inputs expected by EfficientNet networks are the float pixel values of the images between 0 and 255. The minimum image size supported by the EfficientNets is also 32x32.

The InceptionResNetV2 model has some differences when compared to EfficientNets and DenseNets. This network expects a default input shape of (299, 299, 3), and the input data need to be scaled between -1 and 1. 75x75 is the minimum image size accepted by this network.

InceptionV3 network has a default input shape of (299, 299, 3), and the input data scale requirement is between -1 and 1, like InceptionResNetV2. InceptionV3 has a minimum input image size of 75x75.
Table 1. Minimum input shapes of the networks

<table>
<thead>
<tr>
<th>Network</th>
<th>Minimum input shape</th>
</tr>
</thead>
<tbody>
<tr>
<td>DenseNet 121-169-201</td>
<td>32x32</td>
</tr>
<tr>
<td>EfficientNet B0-B7</td>
<td>32x32</td>
</tr>
<tr>
<td>InceptionResNetV2</td>
<td>75x75</td>
</tr>
<tr>
<td>InceptionV3</td>
<td>75x75</td>
</tr>
<tr>
<td>MobileNet - MobileNetV2</td>
<td>32x32</td>
</tr>
<tr>
<td>NasNetMobile - NasNetLarge</td>
<td>32x32</td>
</tr>
<tr>
<td>ResNet 50-101-152</td>
<td>32x32</td>
</tr>
<tr>
<td>ResNet 50V2-101V2-152V2</td>
<td>32x32</td>
</tr>
<tr>
<td>VGG16</td>
<td>32x32</td>
</tr>
<tr>
<td>VGG19</td>
<td>32x32</td>
</tr>
<tr>
<td>Xception</td>
<td>71x71</td>
</tr>
</tbody>
</table>

Source: Authors

The MobileNet and MobileNetV2 networks have a default input shape of (224, 224, 3), and the minimum supported image size is 32x32, like DenseNets. However, these networks require the input data to be scaled between -1 and 1.

NasNets includes NasNetLarge and NasNetMobile, whose input requirements are just like DenseNets, with a default input shape of (224, 224, 3) and a minimum image size of 32x32. As their names explain, NasNetLarge is a larger network, with more parameters to train, and NasNetMobile is the lightweight version of the network, with fewer parameters to train that can however be implemented on more devices.

ResNet networks can be grouped into two: ResNets and ResNetV2s. ResNets have a default input shape of (224, 224, 3), and the minimum input size is 32x32. ResNetV2s also have the same properties, but their input scale requirement is between -1 and 1. This research used ResNet50, ResNet101, ResNet152, ResNet50V2, ResNet101V2, and ResNet152V2.

VGG16 and VGG19 are also networks that have a minimum input size of 32x32 and a default input shape of (224, 224, 3). The last network used was Xception, which has a minimum input size of 71x71 and a default input shape of (299, 299, 3). This network also requires the input data to be scaled between -1 and 1.

All of the networks described above have the option of loading pre-trained weights from a path or from ImageNet. However, in this research, pre-trained weights were not used, which means that the weights were randomly initialized.

Training

For the training process, training parameters were selected after several experiments and used for all the networks. A value of patience parameter was determined in order to track the learning process of the networks, aiming to make sure that the learning process continued as long as the model continued to improve. To this effect, the patience value was set at 100 epochs by experimenting, and validation loss was tracked during the whole training process of each network. This means that each network continued with the training process as long as the validation loss of the network kept decreasing. In other words, the network continued to learn. However, if the network stopped learning and the validation loss did not decrease for the selected patience value of 100 epochs, the training process would automatically stop, and the best weights of the training process would be restored before saving the model. As an example, Xception’s training graph is shown in Figure 3. For comparison, the training of all networks was carried out with the same computer, which had 32GB DDR4 system memory, a 10th Gen. Intel Core i7-10750H processor, and an NVIDIA GeForce RTX 2070 SUPER graphics card with 8GB GDDR6 video memory.

Results and discussion

After the experimental results were obtained, networks with an accuracy above 70% were selected for comparison. ResNet50, ResNet101, and ResNet152, as well as EfficientNets, were not able to obtain a good accuracy with 32x32 grayscale images, so they were not included in the comparison.

After obtaining results with the original dataset for all of the networks, different image preprocessing techniques were evaluated for further improvements to the networks. Contrast stretching was applied to obtain low- and high-contrast images. Furthermore, Discrete Cosine Transform (DCT) was applied, which has improved accuracy by compressing the image in some of the studies. However, in our case, none of these approaches was able to improve the accuracy of the networks.

Since grayscale images with a resolution as low as 32x32 were being used, the task became harder for the networks, even with the original images.
Although image preprocessing affects ANN applications in a good way, our experiments showed that it can negatively affect the results for small input data, which is 1024 (32×32×1) in our research.

Table 3 shows the overall performances of the networks included in the comparison.

By comparing the training times of the networks, it can be seen that VGG16 required the shortest time, with 105 minutes. On the contrary, ResNet152V2 reported the highest value, with 607 minutes. The training times of Xception and MobileNet were also very close to that of VGG16, with 108 and 112 minutes, respectively.

While evaluating the networks on the test dataset, their evaluation were also recorded. An 80-20% train-test ratio was used for all of the networks, and the test dataset included 1,579 images. VGG16 reported the best evaluation time, with 2.52 seconds, and VGG19 was the second best (2.68 seconds). Considering that the image size for Xception was 71×71, its evaluation time of 4.05 seconds was also a good result. The highest evaluation time was 14.2 seconds, obtained by the InceptionResNetV2 with a 75×75 test image size.

The size of the network is also an important parameter for implementation. It may not be always possible to implement a model with the highest accuracy on every device because of the size of the network and its memory requirements. By comparing the output model sizes of all networks, it was observed that the MobileNetV2 network has the smallest size, with 29.6 MB. The MobileNet network has the second smallest size with 38.9 MB, and ResNet152V2 and InceptionResNetV2 have the two largest model sizes (679 and 638 MB, respectively).

The system initialization time refers to the time in seconds required by the face mask detection system for processing during implementation. When the trained models were implemented, the lowest initialization time was obtained by VGG16, with 5.5 seconds, followed by VGG19 followed with 6.1 seconds. The InceptionResNetV2 network reported the highest initialization time (104.2 seconds). In addition, the FPS of the system was also monitored during implementation. The FPS of the system includes both face detection in the captured video and the classification of the detected faces by the network. MobileNet, VGG16, and VGG19 had the highest FPS, with 14. MobileNetV2 and ResNet50V2 followed them with 13 FPS, and InceptionResNetV2 had the lowest value with 8.

Regarding the accuracies of the networks, InceptionResNetV2 and Xception had the highest value, with 99.6%. VGG19 and VGG16 also showed an accuracy over 99 (with 99.4% and 99.1%, respectively). As for the general performance of the networks, if the highest possible accuracy is desired, the Xception network could be selected, as it has moderate values regarding model size, initialization time, and FPS. If a little lower accuracy can be tolerated, albeit still over 99%, VGG16 or VGG19 are also good options, with higher FPS values and the two lowest initialization times. However, they have slightly higher model sizes, which is also a drawback. Figures 3, 4, and 5 are the confusion matrices of VGG16, VGG19, and Xception.

In recent studies on face mask detection, the maximum accuracy obtained from the experimental results has been also used for comparison.

Agarwal et al. (2022) used two different datasets with two classes. They obtained a 99.11% accuracy with a hybrid model with CNNs and SVMs.

Goyal et al. (2022) used a two-class custom dataset and a customized CNN architecture with ResNet-10 based on a Single Shot Detector. They obtained a 98% accuracy in their research.

In a study carried out by by Waleed et al. (2022) a two-class dataset was also used. With a customized deep CNN model, they obtained a 99.57% accuracy.

Aydemir et al. (2022) obtained results in three different cases. The first two cases involved two classes, and the third one used three. The resulting accuracies were 97.49, 100, and 95.95%, respectively.

Guo et al. (2022) used a three-class dataset and an improved YOLOv5 in their study, obtaining a mean Average Precision (mAP) of 96.7%.

Han et al. (2022) used a customized three-class dataset and a network called SMD-YOLO, which is YOLOv4-tiny based. They obtained a mAP of 67.01%.

Eyioz et al. (2021) studied three-class face mask detection and obtained a higher accuracy with an Inception-v3 network (i.e., 98.2%).

Table 4 shows the comparison between these studies and our research. It should also be considered that the input sizes of the networks were lower in this research, i.e., the minimum sizes accepted by the networks.
Table 3. Network performance results

<table>
<thead>
<tr>
<th>Network (Input Size)</th>
<th>Epochs Run</th>
<th>Training time (mins)</th>
<th>Evaluation Time for 1579 test images (s)</th>
<th>Model Size (MB)</th>
<th>System Initialization (s)</th>
<th>FPS</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DenseNet121(32x32)</td>
<td>1 748</td>
<td>226</td>
<td>6,13</td>
<td>89,6</td>
<td>60,9</td>
<td>11</td>
<td>0,969</td>
</tr>
<tr>
<td>DenseNet169(32x32)</td>
<td>1 554</td>
<td>267</td>
<td>6,47</td>
<td>157</td>
<td>77,2</td>
<td>10</td>
<td>0,965</td>
</tr>
<tr>
<td>DenseNet201(32x32)</td>
<td>1 681</td>
<td>334</td>
<td>8,05</td>
<td>224</td>
<td>95,4</td>
<td>9</td>
<td>0,963</td>
</tr>
<tr>
<td>InceptionResNetV2(75x75)</td>
<td>682</td>
<td>205</td>
<td>14,2</td>
<td>638</td>
<td>104,2</td>
<td>8</td>
<td>0,996</td>
</tr>
<tr>
<td>InceptionV3(75x75)</td>
<td>1 810</td>
<td>241</td>
<td>6,95</td>
<td>256</td>
<td>46</td>
<td>11</td>
<td>0,976</td>
</tr>
<tr>
<td>MobileNet(32x32)</td>
<td>3 417</td>
<td>112</td>
<td>3,01</td>
<td>38,9</td>
<td>14,9</td>
<td>14</td>
<td>0,823</td>
</tr>
<tr>
<td>MobileNetV2(32x32)</td>
<td>4 118</td>
<td>198</td>
<td>3,69</td>
<td>29,6</td>
<td>26,7</td>
<td>13</td>
<td>0,730</td>
</tr>
<tr>
<td>ResNet50V2(32x32)</td>
<td>1 445</td>
<td>122</td>
<td>4,55</td>
<td>273</td>
<td>27,6</td>
<td>13</td>
<td>0,898</td>
</tr>
<tr>
<td>ResNet101V2(32x32)</td>
<td>1 808</td>
<td>315</td>
<td>6,73</td>
<td>495</td>
<td>49,3</td>
<td>11</td>
<td>0,882</td>
</tr>
<tr>
<td>ResNet152V2(32x32)</td>
<td>2 173</td>
<td>607</td>
<td>6,83</td>
<td>679</td>
<td>74,2</td>
<td>10</td>
<td>0,880</td>
</tr>
<tr>
<td>VGG16(32x32)</td>
<td>2 078</td>
<td>105</td>
<td>2,52</td>
<td>385</td>
<td>5,5</td>
<td>14</td>
<td>0,991</td>
</tr>
<tr>
<td>VGG19(32x32)</td>
<td>2 412</td>
<td>150</td>
<td>2,68</td>
<td>445</td>
<td>6,1</td>
<td>14</td>
<td>0,994</td>
</tr>
<tr>
<td>Xception(71x71)</td>
<td>587</td>
<td>108</td>
<td>4,05</td>
<td>241</td>
<td>24,4</td>
<td>12</td>
<td>0,996</td>
</tr>
</tbody>
</table>

Source: Authors

Table 4. Comparison with other studies

<table>
<thead>
<tr>
<th>Reference</th>
<th>Classes</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Agarwal et al., 2022)</td>
<td>2</td>
<td>99,11% Accuracy</td>
</tr>
<tr>
<td>(Goyal et al., 2022)</td>
<td>2</td>
<td>98% Accuracy</td>
</tr>
<tr>
<td>(Waleed et al., 2022)</td>
<td>2</td>
<td>99,57% Accuracy</td>
</tr>
<tr>
<td>(Aydemir et al., 2022)</td>
<td>2 (Case 1)</td>
<td>97,49% Accuracy</td>
</tr>
<tr>
<td></td>
<td>2 (Case 2)</td>
<td>100% Accuracy</td>
</tr>
<tr>
<td></td>
<td>3 (Case 3)</td>
<td>95,95% Accuracy</td>
</tr>
<tr>
<td>(Guo et al., 2022)</td>
<td>3</td>
<td>96,7% mAP</td>
</tr>
<tr>
<td>(Han et al., 2022)</td>
<td>3</td>
<td>67,01% mAP</td>
</tr>
<tr>
<td>(Eyiokur et al., 2021)</td>
<td>3</td>
<td>98,20% Accuracy</td>
</tr>
<tr>
<td>This study</td>
<td>3</td>
<td>99,6% Accuracy</td>
</tr>
</tbody>
</table>

Source: Authors

Conclusion

In this research, deep convolutional neural networks were trained with as little input data as possible to obtain an accurate model for face mask detection and wearing condition classification. A dataset was created by using the Labeled Faces in the Wild (LFW) dataset and adding face masks to it in order to simulate correct, wrong, and no mask wearing conditions. According to our experimental results, EfficientNets and ResNet50, ResNet101, and ResNet152 networks were not able to learn with their minimum input size of 32x32 and grayscale images. Four of the trained networks were able to obtain an accuracy of over 99%, i.e., InceptionResNetv2, Xception, VGG16, and VGG19, with accuracies of 99,6, 99,6, 99,1, and 99,4% respectively. VGG16 and VGG19 also had better FPS when compared to other networks.

The results obtained in this study showed a higher performance for three classes while using fewer input dimensions. This, in comparison with other studies in the literature.

Figure 6 shows the output of our application for three different mask wearing situations. Boundary boxes are placed on detected faces, and the classification results are written with their confidence scores.

For future works, an image pre-processing phase will be added to the process, and a custom architecture will be developed to be tested on various different datasets.

CRediT author statement

All authors: conceptualization, methodology, software, validation, formal analysis, investigation, writing (original draft, writing, review, and editing), data curation.

References


Das, A., Ansari, M. S., and Basak, R. (2020). COVID-19 face mask detection using Tensorflow, Keras and OpenCV. In IEEE (Eds.), 2020 IEEE 17th India Council International Conference (INDICON) (pp. 1-5). IEEE. https://doi.org/10.1109/INDICON49873.2020.9342585


