IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 38, No. 6, December, 2021, pp. 1875-1885
Journal homepage: http://iieta.org/journals/ts

Detection of Hidden Facial Surface Masking in Stored and Real Time Captured Images: A Deep Learning Perspective in Covid Time

Ruchi Jayaswal*, Manish Dixit

Madhav Institute of Technology and Science, M.P. Gwalior 474005, India

Corresponding Author Email: ruchi.jayaswal23@mitsgwalior.in

https://doi.org/10.18280/ts.380632

Received: 29 November 2021
Accepted: 20 Decemebr 2021

Keywords:
COVID-19, face mask detection, DNN models, optimizers, CLAHE-SSD_IV3 model, RTFMD dataset

ABSTRACT

A novel coronavirus has spread over the world and has become an outbreak. This, according to a WHO report, is an infectious disease that aims to spread. As a consequence, taking precautions is the only method to avoid catching this virus. The most important preventive measure against COVID-19 is to wear a mask. In this paper, a framework is designed for face mask detection using a deep learning approach. This paper aims to predict a person having a mask or unmask and also presents a proposed dataset named RTFMD (Real-Time Face Mask Dataset) to accomplish this objective. We have also taken the RFMD dataset from the internet to analyze the performance of system. Contrast Limited Adaptive Histogram Equalization (CLAHE) technique is applied at the time of pre-processing to enhance the visual quality of images. Subsequently, Inceptionv3 model used to train the face mask images and SSD face detector model has been used for face detection. Therefore, this paper proposed a model CLAHE-SSD_IV3 to classify the mask or without mask images. The system is also tested at VGG16, VGG19, Xception, MobilenetV2 models at different hyperparameters values and analyze them. Furthermore, compared the result of the proposed dataset RTFMD with the RFMD dataset. Additionally, proposed approach is compared with the existing approach on Face Mask dataset and RTFMD dataset. The outcomes have obtained 98% test accuracy on this proposed dataset RTFMD while 97% accuracy on the RFMD dataset in real-time.

1. INTRODUCTION

The novel coronavirus has spread to numerous countries throughout the world, causing a pandemic. More than 216 countries have been afflicted by a coronavirus, according to a World Health Organization report [1, 2]. It's also known by other names like Coronavirus, Corona, Novel Coronavirus, and Covid-19. According to a WHO study, this novel virus was originally identified in December 2019 in Wuhan, China, and has since caused a global pandemic; as of June 21, 2020, more than 8.75 million cases had been reported worldwide, resulting in more than 463,000 deaths [1]. COVID-19 is a respiratory infection that causes an infectious sickness. Fever, cough, exhaustion, breathlessness, sore throat, headache, runny nose, and other symptoms are prevalent. Some people get infected but only experience minor symptoms.

The virus can spread before the symptoms occur, if someone comes into contact with someone who does not display symptoms in the early days. Even so, if people do not take the necessary precautions, the infection can spread. In addition, one of the major factors of the virus's propagation is travel.

However, there are some vaccines discovered to prevent this virus. But they are not much effective because numbers of covid variants spread. Though, WHO has recommended some protective measures to reduce this fatal virus's chances [3]. Some of the precautions can assist you avoid becoming infected. Covering the face with a mask (particularly when talking to someone or in a public location with a large number of people), often washing hands, sanitising hands, avoiding touching the face repeatedly, and maintaining social distance are all ways to prevent the spread of viruses. Another safety measure implemented by the Indian government is the Argoya Setu App, which allows individuals to notify authorities in advance if they come into contact with an infected person or are within the infected person's radius, and having this app on their phone is a requirement. Figure 1 depicts the global scenario in terms of the number of instances increasing without any precautions or safety measures applied and reducing when safety measures are taken.

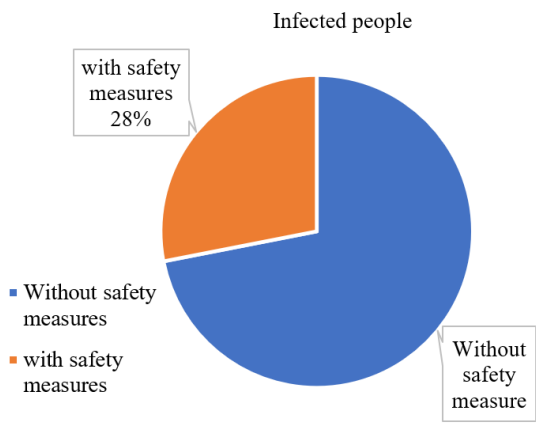

Figure 1. COVID-19 spread over the world, with or without precautions

In context of the present circumstances, or to put it another way, following the lockdown period, WHO or governments advise people to put on a mask and keep social distance [2]. This coronavirus is not transmitted to another person with the use of a mask, while others suffer from colds and coughs. Furthermore, several service providers require customers to wear masks in order to enter and use the service.

Artificial Intelligence is important when machines can do tasks that need human intelligence. AI is a superset of machine learning and Deep learning is a subset of machine learning where the machine behaves like human brain and learn from huge amount of data.

Hence, face mask detection becomes an essential computer vision task to help society, and also preliminary researches have been done in face mask detection. This paper will discuss face mask detection techniques based on deep learning. Face mask detection refers to detecting whether a person wears a mask or not and mentions the face's location by using a boundary box [3]. This face mask detection is implemented on proposed Real-Time Face Mask Dataset (RTFMD). At this time, a limited number of databases are available on the internet to detect the mask.

We have designed a new face mask dataset RTFMD and will explain it in a further section. In this work CLAHE technique is used to improve vision quality of images and applied data augmentation for increase the size of dataset. Further, Inceptionv3 model is used to train the face mask dataset by tuning the hyper-parameters. At the testing time, firstly SSD face detector model is applied to detect faces from streams or images then obtained trained face mask model which is used to classify the face without mask or with mask. Further, we will see CLAHE-SSD-IV3 approach is the best suitable approach among other models for face mask detection. The experiment will be shown of VGG16, Xception, MobileNet, VGG19 and CLAHE-SSD_IV3 models and compare all of them. By applying proposed CLAHE-SSD_IV3 model, the face mask detection framework gives high accuracy in real-time. Later the comparison of existing dataset and proposed dataset will be discussed. The comparison between the recent research and suggest research on face mask detection is also discussed in section 5.

1.1 Motivation and challenges

Face mask identification in real time has become a new issue for researchers. Furthermore, only a few algorithms are developed to deal with this problem in a real-time environment. A competent face detection system that can detect faces in real frames is required to detect the face mask. In a pandemic scenario, a trustworthy dataset and efficient algorithm are required to create a strong face mask detection system. This paper's purpose is to address these two issues. First, we introduced the RTFMD dataset and proposed the CLAHE-SSD_IV3 technique, which is a very fast model for training and testing the dataset. In this research, the accuracy of an existing dataset, RFMD, will be compared to the novel RTFMD dataset.

This paper's structure is divided into seven sections. Section II: Literature Review will clarify and relate to the previous work. The proposed face detection methodology will be discussed in Section III. The proposed dataset will be summarized in Section IV. The experiment, evaluation metrics and analysis of results will be discussed in section V. The analysis will focus on two datasets, and on the proposed method. The conclusion will be discussed in Section VI.

2. LITERATURE SURVEY

Jiang et al. [4] proposed "RetinaFaceMask," a technique for detecting face masks that is efficient and accurate. He presents a new context attention module strategy for recognizing face masks and also proposed a cross-class object elimination approach that excludes low confidence predictions as well as the union's high intersection. The MobileNet and ResNet models were employed in this strategy to detect whether there was a mask or not. The precision and recall of this strategy to detect mask using MobileNet + RetinaFaceMask are 82.3 percent and 89.1 percent, respectively, and 93.4 percent and 94.5 percent when using ResNet and RetinaFaceMask.

A mask face dataset was proposed by Wang et al. [5]. He recommended three face mask datasets, all of which are open source. There are two components to the Masked Face Detection Dataset (MFDD). Some of the images were culled from the Internet, while others came from relevant study. The mask pictures in MFDD total 24,771. The second dataset is called the Real Mask Face Recognition Dataset (RMFRD). There are 5000 images of 525 people wearing masks and 90000 shots of the same 525 people without masks in this collection. The third dataset is SMFRD (Stimulated Mask Face Recognition Dataset) and used masks on existing public face datasets to create this dataset. In addition, 500,000 face photos of 1000 persons were included in this collection.

Ge et al. [6] took on the face detection problem and produced the MAFA dataset, which included 30,811 internet photos and 35,806 masked face images. The images in this collection have varying degrees of occlusion, and at least one section of each Face has been corked by mask and orientation. Furthermore, LLE-CNNs with three primary modules are presented for face mask detection. The first module uses two pre-trained CNNs to extract face traits from input photos and treat them as a high-dimensional descriptor and these descriptors are transformed into similarity-based descriptors. Subsequently, the dictionaries were trained on a large number of synthetic normal faces, masked faces, and nonfaces. Finally, the verification module uses a unified CNN to recognize the candidate's face areas by integrating classification and regression tasks. The proposed approach, outperforms six states of the arts by at least 15.6 percent.

RetinaFace, a powerful single-stage face detector that performs pixel-wise face localization and takes advantage of both extra supervised and self-supervised multi-task learning, was presented by Deng et al. [7]. The paper is divided into five sections. In the first section, the marked five facial landmarks on the images and observed that adding an extra supervision signal improved hard face detection. Subsequently, a self-supervised mesh decoder branch is added to forecast pixel-wise 3D shape face information. RetinaFace then improved the state-of-the-art average precision by 1.1 percent on the WIDER Face hard test set in the third stage. RetinaFace's fourth stage allows state-of-the-art approaches to improve their face verification outcomes (TAR=89.59 percent for FAR=1e-6). Finally, in the final module, RetinaFace may run real-time on a single CPU core for a VGA resolution picture using lightweight backbone networks.

Zhang et al. [8] introduced AInnoFace, an effective technique for face detection. The methodology begins with one step, RetinaNet, and includes certain performance-

enhancing techniques. For regression, he used the Intersection over Union loss function, for Face detection, he used two-step classification and regression, and the data augmentation revisited that based on data anchor sampling used for training purposes. For classification, the max-out operation and a multiscale testing technique is used. As a result, on the WIDER face dataset, the method enhances face detection performance.

3. PROPOSED METHODOLOGY

In this article, the proposed methodology introduces the CLAHE-SSD_IV3 approach. This section discusses utilizing deep learning models to recognize a face mask. The system uses a technique to predict if persons have properly worn a mask; the firstly (pre-processing and training) model is trained on proposed RTFMD dataset and existing RFMD dataset. Brief detail about the RTFMD and RFMD dataset have been discussed in section 4.1 and 4.2. The pre-processing is done before training. Resized all images and Contrastive Limited Adaptive Histogram Equalization (CLAHE) technique is used to enhance the quality of images. Subsequently, image augmentation is applied to increase the size of RTFMD dataset and further build a model. After training, phase II: a face detection model is required to recognize faces so that the CLAHE-SSD_IV3 model can determine whether the person is wearing a mask. The goal of this study is to increase the mask detection system's accuracy in less time while simultaneously maintaining its reliability across all datasets. Figure 2 shows a flow diagram of the approach used in this paper. At the benchmark, many models are used to compare with the proposed approach, and possible hyperparameters are changed to validate the results. In this part, we'll go through these terms briefly.

The stepwise approach of proposed CLAHE-SSD_IV3 methodology has been described as shown below. Firstly, the face mask datasets are loaded as an input. Resize function and CLAHE pre-processing technique are applied to convert the images into compatible format of model and raise the quality of image. Further, Image augmentation technique is applied in order to improve accuracy. Image data are then splitted into training (70%) and validation (30%) batches randomly using train and test_split function. Subsequently, apply a transfer deep learning model to extract the features of face mask images. An Inceptionv3 model is used to learn the features. Moreover, the Inception model is fine-tuned as per the model's requirements and obtained accuracy and computation time for training and validation set. Furthermore, the trained face mask model obtained during training was then deployed at the time of testing and apply Single Shot (SSD) detector face detector model to detect the face within the frame using the boundary box. Then the face mask detector model is used to classify 'with mask' or 'without mask' faces by marking the boundary box. The face mask detection system is tested on both stored and real time images.

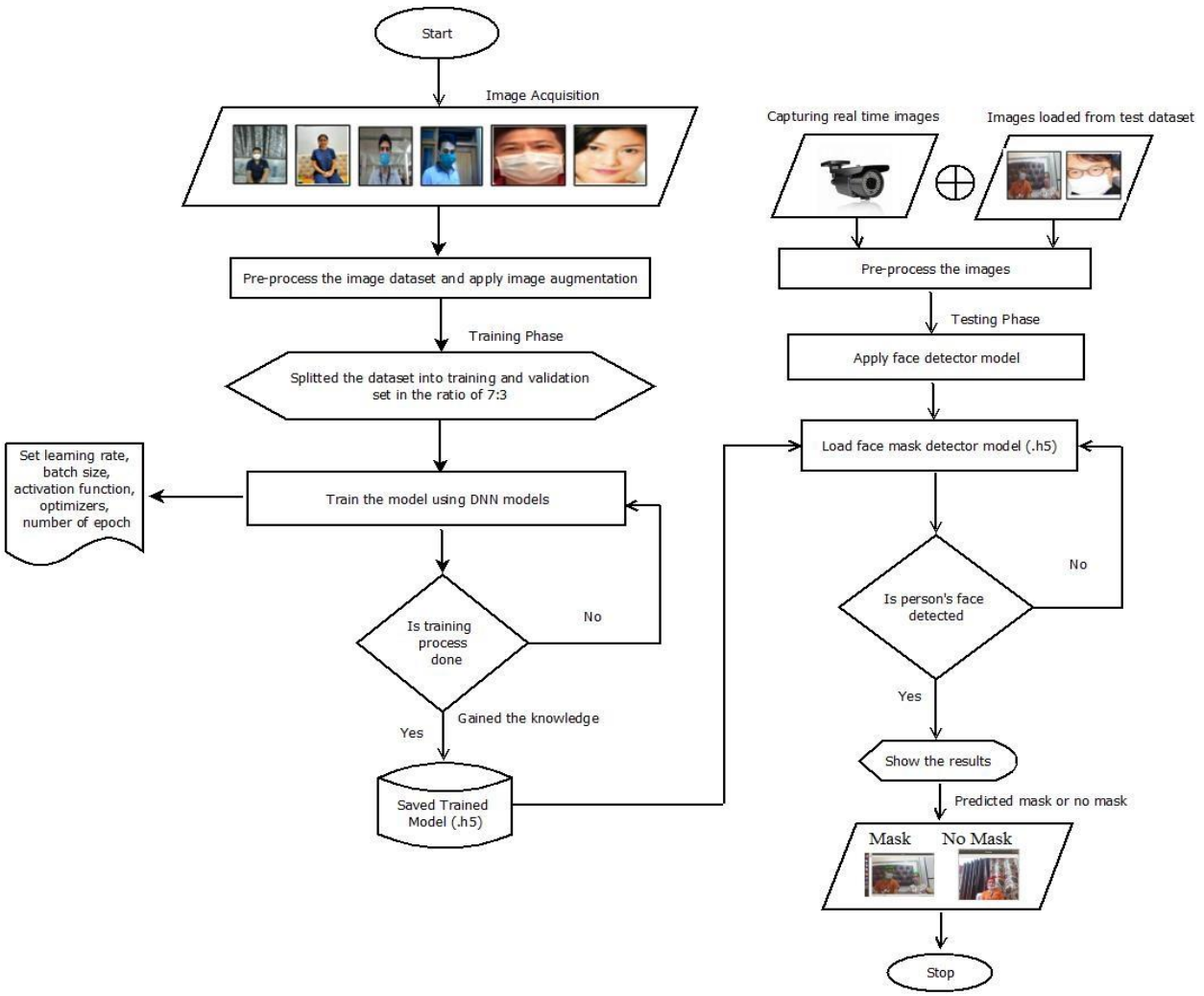

Figure 2. Flowchart of face mask detection system

Pseudo code for Face Mask Detection System

Input: Load image dataset (RTFMD), each picture is captured 2 to 6 feet distance from camera, Image size (a*b*c).
Output: Predict a person having "mask" or "no mask"
Step 1: Acquired required images from camera at a 2 to 6 feet distance.
Step 2: Load face mask dataset from disk.
Step 3: Apply Resizing and CLAHE techniques on dataset.
Step 4: Apply Data Augmentation technique and then split data randomly into 70% training and 30% validation batches.
Step 5: Train images with InceptionV3 model (size= (299,299,3), along with tune the hyper-parameters and optimizer functions.
Step 6: Obtained and save the face mask detector model (.h5) and check accuracy of model and time elapsed of generating model.
Step 7: Capture images for testing from webcam or load images from stored database.
Step 8: Deploy pre-trained SSD face detector model,
    If people face is detected from the static image or from real time frames.
        Crop the face by bounding box coordinates.
Step 9: Deployment of face mask detector model(.h5) and print accuracy,
    Get the prediction from the mask detector model
        If person wear a mask
            then print "with mask" on boundary box
        Else
            then print "no mask" on boundary box
Step 10: End real-time stream when "q" is clicked.

3.1 Pre-processing and data augmentation of dataset

The pre-processing of the dataset is the first stage in the methodology, and it involves resizing the image dataset to fit the suggested model. Because there are few publicly available face mask datasets, we gathered data with and without face mask images. The detailed description of the dataset will be discussed in section 4.1; thus, all images must be resized to a specific resolution. Because the RFMD dataset is so huge, only 3000 images were chosen for assessment. This dataset will be discussed in length in section 4.2, with all images being carefully selected and resized. The list of photos is converted to a NumPy array after resizing for easier and faster calculation.

3.1.1 CLAHE method

Contrast Limited Adaptive Histogram Equalization (CLAHE) [9] is a variation of adaptive histogram equalization in which contrast amplification is constrained in order to minimize noise amplification. In CLAHE, the contrastive limited technique must be employed for each neighborhood from which a transformation function is derived. Rather than recording the complete image, CLAHE reduces overamplification by dividing it into little data portions known as tiles, which are then contrast increased. After that, the tiles are combined to form a more comprehensive image. It can be applied to both grayscale and colour images. Figure 3 depicts the sample image of face mask images after applying CLAHE method.

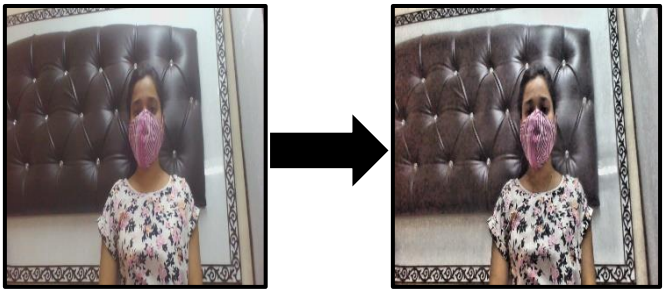

Figure 3. Sample image after CLAHE filter

3.1.2 Image augmentation technique

The image augmentation method is used to increase the training model's accuracy. A massive amount of data is required to train deep learning models for greater accuracy. The CLAHE-SSD_IV3 model is trained in this proposed framework after image augmentation of the dataset. Because the RTFMD dataset is insufficient for training, the augmentation strategy is particularly effective for improving performance. To enhance the size of our suggested dataset, we can use augmentation methods such as rotation, flipping, shearing, shifting, brightness, and zooming. This approach is used in both datasets to achieve superior results. The image data generation function is used for data/image augmentation, and it also returns image train and validation batches.

3.2 Face detector SSD model

Face detection [10] is a critical phase in this framework. The initial stage of the face mask detection system is to detect faces in stored and real-time images. When a person's face appears in front of the camera during testing, it is critical to first identify the face and then establish whether or not the person is wearing a mask. The Caffe Single Shot Multibox Detector (SSD) model, which has been pre-trained, is used to detect the face. Its backbone is based on the ResNet-10 architecture. The Caffe model is part of a deep neural network module added in OpenCV 3.3. To detect the face, two files are loaded: Caffemodel and prototxt files.

Faces are detected by constructing boundary boxes of the face after applying the Caffe model with cv2.dnn.readNet ("path/to/prototxtfile","path/to/caffe-weights").

Faces are detected in real time frames using the Caffe model, even in varied face orientations such as left, right, top, and bottom.

3.3 Classification of images using Fine-tuned InceptionV3 architecture

Figure 4 depicts the core architecture of Inceptionv3. The InceptionV3 model is used to extract features [11] and classify image datasets. More improvements are included in this

network design, such as the RMSProp optimizer, factorized 7*7 convolutions, batch normalisation, and label smoothing [12-14]. This network came in second place in the picture classification competition at ILSVRC 2015 [12, 13]. To fine-tune the model, we create it and load it with ImageNet weight, then freeze the base layers and conduct top layer truncation by defining another fully connected (FC) softmax, unfreeze the head layer, and replace it with the new FC layer. A classification layer is used to assess the performance of a fully connected layer based on its anticipated probability. The Softmax function is utilized for categorizing the multiple divisions in classification layer [15]. The function's formula of softmax is provided in Eq. (1); it takes the CNN result as a vector of score z and compresses it in the range of 0 to 1, with the sum of total equal to 1 [16]:

$$S_n(z) = \frac{e^{z_n}}{\sum_{i=1}^{N} e^{z_i}} \quad (1)$$

Cross entropy (L) is then calculated as shown in Eq. (2):

$$L = -S_{y_i} + log\left(\sum_{n=1}^{N} e^{S_n}\right) \quad (2)$$

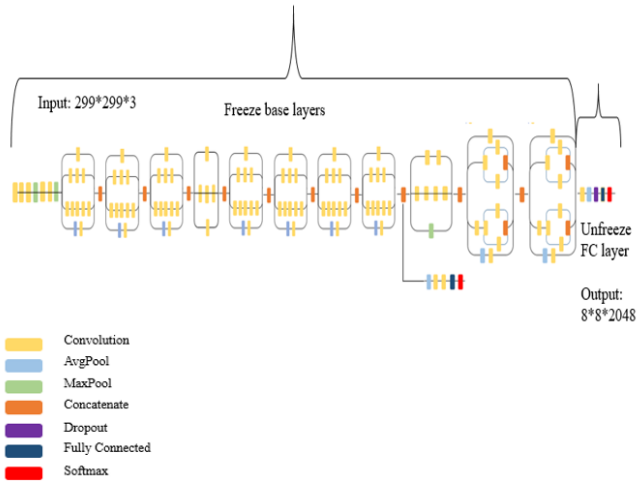

Figure 4. Fine-tuned Inceptionv3 architecture

We got high dimensional features using CNN since it can extract high-level features at a minimal computational cost, as well as train deeper networks. To validate the accuracy of the RTFMD dataset, this system is additionally trained using MobilenetV2 [16], VGG16 [17], VGG19 [2], and Xception [18] models with varied tuning of hyperparameters, as shown in Table 1. These models are ready for architecture fine-tuning [19]. Fine-tuning is an important step in improving the accuracy and speed of the training data. One sort of transfer learning [20] that can outperform the feature extraction method is fine-tuning [21]. It creates a new fully-connected head using this tuning process. Freeze the basic layers to prevent them from being changed in future training sessions. The head layer weights, on the other hand, will be tweaked. Further, examine the accuracy and of all the models and as well as the system's overall performance.

3.3.1 Optimizer

An optimizer is required to reduce the error rate of a deep learning model. The optimization algorithm contributes to the accuracy and speed with which a model is developed. Optimizers are an improved class that contains more parameters to train a model. Some of the optimizers that can be used to improve the results are SGD [22], AdaGrad [23], RMSProp [23], ADAM [24], ADADELTA [25], Nesterov [26], and Ftrl [27] and each has benefits and drawbacks. The ADAM optimizer is used in conjunction with the provided technique, and the remaining results are compared and analysed in Table 3.

Adaptive Moment Estimation: This optimizer is gaining traction and is useful in deep learning applications. It was created by integrating the momentum and RMSprop techniques. The updating technique is identical to RMSprop, except that a smooth version of the gradient is used instead of raw stochastic gradients. Adam also has a bias adjustment. It's a straightforward approach that uses less memory space [24]. Eq. (3)-(6) illustrate the updating of the gradient and include a bias correction strategy for each parameter w^j.

$$v_t = \beta_1 * v_{t-1} - (1 - \beta_1) * g_t \quad (3)$$

$$s_t = \beta_2 * s_{t-1} - (1 - \beta_2) * g_t^2 \quad (4)$$

$$\theta = -\alpha * {v_t}/{\sqrt{s_t + \epsilon}} * g_t \quad (5)$$

$$w_{t+1} = w_t + \Delta w_t \quad (6)$$

where, α is a learning rate, g_t (gradient at time t along w^j), v_t equivalent to exponential average of gradients along w^j. The exponential average of square of gradient along w^j is s_t and here β_1, β_2 are the hyperparameters.

For each parameter, we compute the gradient's exponential average as well as the squares of the gradient (Eq. (3), and Eq. (4)). We multiply the learning rate by the average of the gradient (as with momentum) and divide it by the Root Mean Square (RMS) of the exponential average of square of gradients (as with momentum) in Eq. (5) to determine the learning step. The update is then added. The hyperparameter β_1 is usually set to 0.9, while β_2 is set to 0.99. In most cases, ϵ is set to 1e-10.

4. MATERIAL REQUIRED

4.1 Proposed dataset: RTFMD

We proposed a new dataset called Real-Time Face Mask Dataset to determine whether or not a person is wearing a mask (RTFMD). RTFMD features 620 images, 330 of which have masks while the remaining 290 without mask. They're all taken with a webcam or a mobile phone camera, and the distance between the camera and the subject ranges from 2 to 7 feet. A random set of pictures was taken with no consideration for gender. To train with detecting masks in this dataset, many types of masks which includes 2d printed masks, handkerchiefs, towel masks were worn. RTFMD supports three types of image set. One set of person appears in one picture with a mask, another set of people appears without a mask image set, and last of two or three persons appear in one picture with or without a mask. In each image, several lighting approaches and positions were applied (including side poses and front side). Figure 5 shows the sample images of RTFMD dataset.

Figure 5. The sample of RTFMD dataset

4.2 RFMD

The second dataset, the Real World Masked Face Dataset (RFMD) [5] was downloaded from the internet and contains two types of images: masked and unmasked faces. There were 5000 veiled faces of 525 persons and 90,000 uncovered faces in all. These pictures were gathered at train stops and other locations after being crawled from the internet. However, we only took 3000 photographs, 1500 of which feature mask faces and 1500 of which without mask. Figure 6 represents the sample images of RFMD dataset.

Figure 6. Sample of RFMD dataset with mask and unmask faces

5. EXPERIMENTAL SETUP, EVALUATION METRICS AND RESULT ANALYSIS

The experiment setup, assessment measures, and outcome analysis are discussed in this section. To assess the model's accuracy, various models with varying hyperparameter values are investigated. To examine the system's performance, hyperparameter values are fine-tuned. A hyperparameter is a parameter whose value can be set before to the start of the learning process to achieve the best results. Because the values of hyperparameters varied with the new application [28], the setup of hyperparameters values is humane. The task of picking a set of ideal hyperparameters for a machine learning or deep learning algorithm is known as hyperparameter tuning [29]. Learning Rate (α), Batch Size (m), Epochs, Activation functions, and Optimizers are used to assess the system's performance in this article. The outcomes vary depending on the hyperparameter setting.

5.1 Experiment setup

In this experiment, we used a laptop with 16 GB RAM, an i7 core processor, and 4GB NVIDIA GPU processor, as well as Ubuntu OS. The models are trained after the implementation work is completed on the Spyder platform. With 70% and 30% images, respectively, then divided the dataset randomly into a train and validation set. We've also trained several models without using a GPU, but it takes longer. As a result, efficiency suffers. As a result of switching to a GPU environment, training time increased significantly.

5.2 Evaluation metrics

Precision, recall, accuracy, and loss or error rate were used as metrics in this system [30, 31], and they were expressed as follows:

Precision (PR) is described as calculating accuracy and measuring the percentage of relevant outcomes in the test window from correctly recognized results.

$$\mathrm{PR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{7}$$

Recall (RC) is the proportion of the number of relevant query returns to the number of images in the dataset.

$$\mathrm{RC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{8}$$

Accuracy (AC) is defined as the ratio of classified correctly classes to total testing classes in a dataset.

$$\mathrm{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{9}$$

where, TP= True positive class, TN=True Negative class, FP= False Positive class and FN=False Negative class.

Binary Cross-Entropy Loss: It is also known as Sigmoid Cross-Entropy Loss and used when the binary classification problem is seen (C=2).

$$\begin{aligned} CL = -\sum_{i=1}^{c=2} t_i \log\big(f(s_i)\big) \\ = -t_1 \log\big(f(s_1)\big) - (1 - t_1)\log\,(1 \\ - f(s_1)) \end{aligned} \tag{10}$$

where, CL= crossEntropy loss, t_1=1 means that the class C1=C_i is present, s_i is the gradient with respect to each score.

5.3 Result analysis

Table 1 summarizes the performance of the face mask detection model. In this case, the number of epochs used to train the model is 30, and Table 1 summarizes the efficiency of the face mask detection model. The model was built across 30 epochs, and all used distinct models have 256 dense layers; the activation function is Sigmoid, and the Adam optimizer is utilized. The learning rate is an important parameter to tune in CNN models or machine learning algorithms. Choosing a proper scale to determine the learning rate, as well as the batch size during the learning process, is crucial. Table 1 shows how the training model's learning rate and batch size effect accuracy and time. The SSDIV3 model had a better average accuracy on the RTFMD dataset. The accuracy of a model is affected by tuning its parameters. Tables 1, 2 and 3 show the results of a comparison of various models with the proposed CLAHE-SSD_IV3 on the RTFMD dataset.

Table 1. Impact on accuracy and time elapsed of models by changing learning rate and batch size

S.No.	Name of Models	Accuracy(%) and Time(ms) Learning Rate(α) and Mini Batch Size(m)								
		0.0001 16	0.0001 32	0.0001 64	0.001 16	0.001 32	0.001 64	0.01 16	0.01 32	0.01 64
1	MobilenetV2	91%, 152.9 ms	92% 154.9ms	92% 146.1ms	91% 153.7ms	93% 151.2ms	91% 146.4ms	92% 150.2ms	90% 145.6ms	92% 146.7ms
2	VGG16	70% 163.1 ms	72% 165.2ms	68% 152.8ms	86% 163.6ms	86% 155.2ms	80% 152.8ms	88% 163.9ms	80% 155.6ms	80% 152.8ms
3	VGG19	74% 170.3ms	71% 159 ms	64% 156 ms	83% 172.7ms	82% 160.5ms	78% 184.2ms	81% 171.80ms	84% 161.92ms	81% 160.2ms
4	Xception	93% 296.7ms	94% 281.6ms	93% 280.4ms	94% 290.05ms	96% 281.29ms	94% 274.591ms	96% 282.61ms	95% 276.43ms	97% 269.62ms
5	CLAHE-SSD_IV3	96% 274.58ms	96% 267.89ms	94% 267.13ms	98% 270.76ms	96% 266.6ms	98% 272.93ms	96% 276.27ms	96% 276.35	97% 271.09ms

Table 2. Impact on accuracy and time elapsed of models by changing activation function, dense layer and epochs

S.No.	Name of models	Accuracy (%) and Time(ms) Softmax Activation Function and DL=256, n=30, α=0.01, m=64	Number of Dense layer is 512 and AF=sigmoid, n=30, α=0.001, m=64	Number of epochs are 50 and DL=256, AF=sigmoid, α=0.001, m=64
1	MobilenetV2	93%, 145.6ms	93%, 144 ms	93%, 227.9 ms
2	VGG16	85%, 158.7ms	81%, 148 ms	90%, 253.2ms
3	VGG19	81%, 156.1ms	80%, 158 ms	85%, 258.39ms
4	Xception	97%, 271.6ms	94%, 274.9 ms	95%, 454.8ms
5	CLAHE-SSD_IV3	97%, 260.8ms	96%, 261.4ms	97%, 432.2ms

Table 3. Impact on accuracy and time elapsed of models by changing optimizers

S.No.	Name of models	Accuracy(%) and Time(ms) Several Optimizers and DL=256, AF=sigmoid, α=0.001, m=64, n=30 SGD	RMSProp	AdaGrad	Adam	AdaDelta	NAdam	Ftrl
1	MobilenetV2	92%, 192.8ms	94%, 184.2ms	91%, 187.7ms	95%, 146.4ms	54%, 185ms	96%, 188.4ms	89%, 185.1ms
2	VGG16	61%, 250.2ms	85%, 227.7ms	58%, 227.5ms	58%, 152.8ms	53%, 228.1ms	85%, 229.1ms	53%, 227.9ms
3	VGG19	61%, 244.8ms	79%, 243.4ms	57%, 244.1ms	78%, 184.2ms	38%, 243ms	79%, 244.5ms	53%, 243.3ms
4.	Xception	91%, 427 ms	95%, 429 ms	89%, 431.6ms	95%, 274.1ms	71%, 426.9ms	93%, 428.9ms	53%, 430ms
5.	CLAHE-SSD_IV3	96%, 365.5ms	92%, 354.0ms	92%, 350.3ms	97%, 272.3ms	70%, 351.6ms	95%, 354.4ms	76%, 343.8ms

Table 2 shows how the model performs when the activation function (AF), number of layers - dense layer (DL), and epochs are varied (n). The training performance fluctuates as the hyperparameter values are modified. The activation function helps to retain useful info while suppressing irrelevant data. As seen in the table, Sigmoid is used for binary classification and is a good output layer, whereas Softmax is used for multi-classification applications. Dense layer neurons can be modified in hidden layers, but they cannot be changed in layers. Epochs have a significant impact on the training model's success. Overfitting will occur if the number of epochs is set too high. The accuracy of validation data for each iteration to assess if its over-fits the training data or not. These hyperparameters perform well on proposed model.

Table 3 shows a variety of optimizers that will be used to examine the performance of our models. Figure 11 graph depicts the results of a comparison of five optimizers on different models. In comparison to their optimizers, ADAM and NADAM perform well on our dataset. The AdaDelta and Ftrl optimizers do not produce unsatisfactory results. When compared to AdaDelta and Ftrl optimizers, SGD, AdaGrad, and RmsProp produce better results. Overall, the suggested model and the Xception model are more accurate, however the Xception model takes longer to compute than the suggested model. The experimentation comparison of all models with proposed model has been shown in Table 1, 2 and 3. Majorly, in terms of accuracy, the CLAHE-SSD_IV3 model has achieved best accuracy on novel dataset, because the proposed model extracts the pertinent features for all images and perform better classification.

Figures 7-10 show model training accuracy and loss, as well as model validation accuracy and loss. Other models, in addition to the proposed model, were employed to train the dataset, test accuracy, and conclude that the CLAHE-SSD_IV3 model performed better than the others. Overfitting occurred in Xception and Mobilenetv2 models, indicating that the models were overtrained, but not in suggested model. Since, proposed model extracts features very deeply along with useful and for all of the experiments, we set the patience parameter to 10 and 15, which means that the model will automatically stop training after 30 and 50, iterations respectively, if there is no improvement in validation losses. This prevents the model from becoming overfitted. However, there is a continuous progress in validation loss in proposed

model while in other models no such improvement occurred. Overfitting also depends on the model's layers therefore for RTFMD dataset proposed model works well. VGG16,19's train and validation accuracy has been poor. MobilenetV2 and Xception models produce more validation loss. Based on the statistics, we can conclude that Xception produce good validation accuracy but low validation loss, however, CLAHE-SSD_IV3 takes less time to compute the build the model along with high validation accuracy with low validation loss as compared with Xception model. As a result, we chose the CLAHE-SSD_IV3 model for the face mask detection system for this proposed study. Table 2 shows how the model performs when the following hyperparameters are changed: activation function (AF), number of layers, dense layer (DL), and epochs(n).

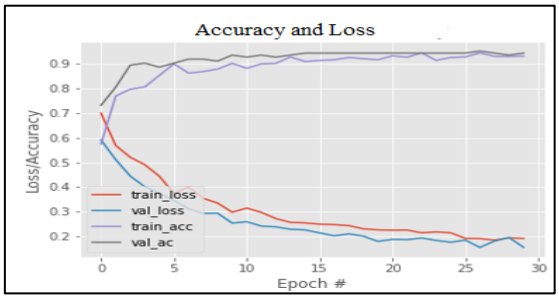

Figure 7. SSDIV3 model

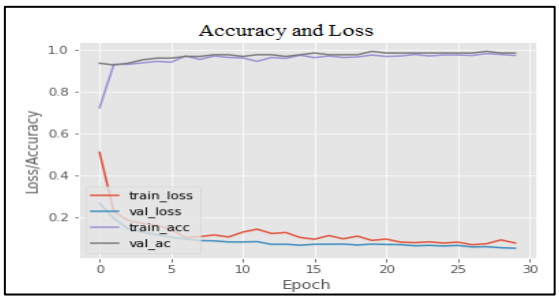

Figure 8. Xception model

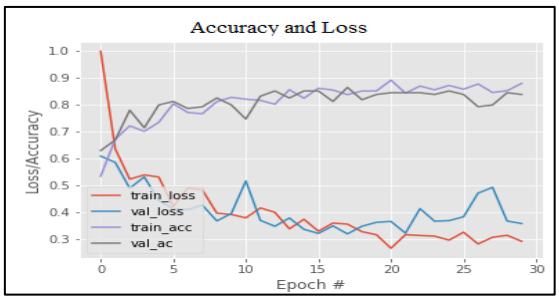

Figure 9. MobilenetV2 model

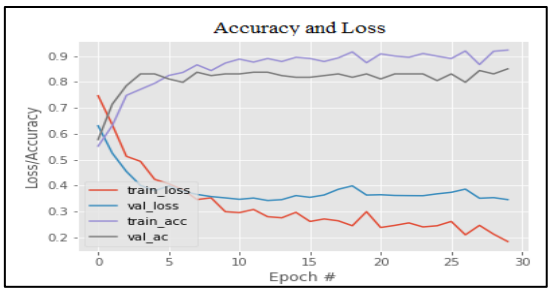

Figure 10. VGG19 model

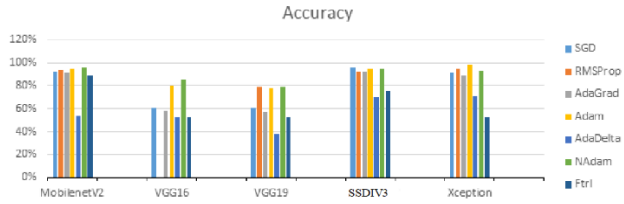

Figure 11. Graph shows the accuracy comparison of seven optimizers in five models

Based on the results of the aforementioned experiments, we can conclude that models performed well when the learning rate=0.001, batch size= 64, set number of dense layers at 256, number of epochs=30, sigmoid classification function and ADAM optimizer were used. The Figure 12 depicts the performance of all models with considering all above parameters. It shows the trade-off between accuracy and time elapsed of models and presents a comparison among all of them.

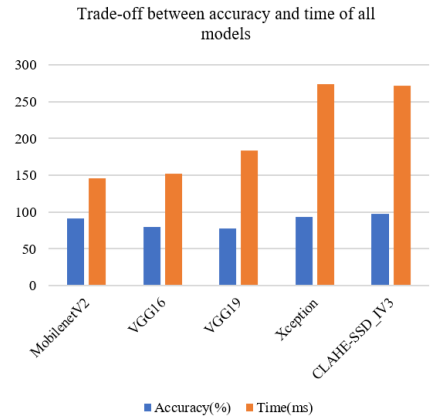

Figure 12. Graphs shows the trade-off between accuracy and time elapsed of all models

5.3.1 Results at run time by using RTFMD trained model

Figure 13-16 shows the real time testing images of people who wear masks or without masks using trained proposed model on RTFMD dataset. The test accuracy on the RTFMD dataset is better than the RFMD dataset at run time. The findings of images using proposed trained model on RFMD dataset, i.e., a person wearing a mask at run time, is shown in the Figures 17 & 18. The system predicts that the person will wear the mask if it simply covers the mouth; however, it should not be depicted in Figure 18. As a result, this is a false positive; the error rate when employing the RFMD trained model is significant. The RTFMD dataset is more reliable and resilient for the proposed method.

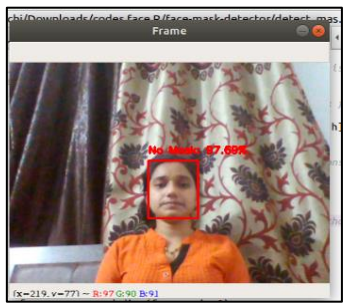

Figure 13. Predict with no mask

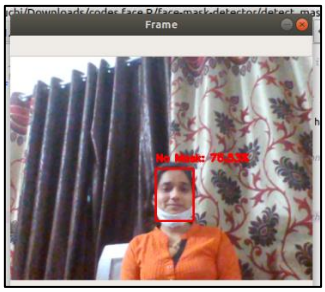

Figure 14. Predict no mask while cover mouth

Figure 15. Predict mask with changes of face direction in real time

Figure 16. Predict mask or no mask, loading examples from disk

Figure 17. Predict mask at run

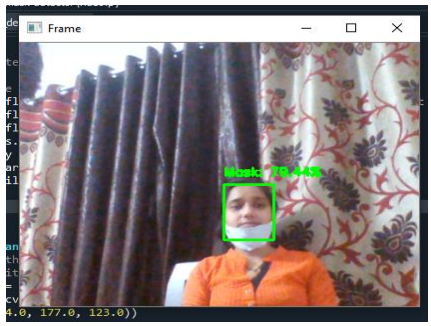

Figure 18. Predict mask while not wearing mask

5.3.2 Results at run time by using RFMD trained model

Now, as indicated in below Table 4, we have experimented with the RFMD dataset and compare its performance with the proposed dataset. To train the model, the CLAHE-SSD_IV3 model is employed. The learning rate is set to 1e-3, the batch size is set to 64, and the epochs are set to 30. In this table, the activation function is Sigmoid, the dense layer is 256, and the ADAM optimizer is used to test the system's performance on RTFMD and RFMD datasets.

Table 5 compares the proposed system to other current models and datasets. The author [4] proposed a two-model RetinaFaceMask with ResNet models using a transfer learning technique on the Face Mask dataset. The author [32] suggested a MobileNet model and found the accuracy for face mask datasets. In addition to these, author [33] present a sequential CNN model and estimate the accuracy. Furthermore, the proposed approach is used to evaluate the accuracy of model on novel dataset and existing face mask datasets.

Table 4. Train and Test Accuracy on both datasets using proposed model

Dataset Name	Total number of images	Proposed Model Name	Train Accuracy	Test Accuracy at run time
RTFMD (Proposed dataset)	**620**	CLAHE-SSD_IV3	**98.77%**	**98%**
RFMD	3000	CLAHE-SSD_IV3	98.85%	97%

Table 5. Comparison of proposed dataset and model with another dataset and model

Approaches	Accuracy (%)		
	Existing Face Mask datasets		Proposed Face mask dataset
	Face Mask Dataset [4]	Medical Face Mask dataset [33]	RTFMD
Jiang [4]	93.49	94.29	95.45
Nagrath [32]	92.48	93.02	93.25
A.Das [33]	94.58	93.29	96.78
Proposed model	**95.96**	**96.82**	**98.12**

6. CONCLUSIONS

The face mask detection application was analyzed in this paper. We used two datasets: RTFMD (Real-Time Face Mask Dataset), which we created with webcams and smartphone cameras, and RFMD (Real World Masked Face Dataset),

which we obtained from the internet. Because of the shifting illumination effect, positions, distinct masks, and several people in one frame, the RTFMD dataset was chosen to evaluate the robustness of the proposed technique. The suggested work's method began with image resizing, and the CLAHE approach was employed to improve image quality. Following that, the InceptionV3 model is used to build or train the model with tuned hyperparameter values such as learning rate, batch size, epochs, dense layer, and activation function, and then the accuracy and time elapsed are measured. These hyperparameters have an effect on model outcomes. Four distinct models were utilized to evaluate system performance and total time consumption. Furthermore, face detection was accomplished with the aid of a pre-trained SSD Caffe model. Moreover, after deploying the trained face mask model, the test accuracy is achieved. When compared to previous models, the suggested model outperformed in terms of accuracy and elapsed time. The performance of these models tested after they were trained using seven different optimizers. On the RTFMD dataset, ADAM and NAdam both performed well. Finally, we were able to achieve 98 percent RTFMD train and test accuracy. Using the RFMD dataset, we trained the CLAHE-SSD IV3 model and subsequently tested its accuracy, reaching 97 % at test time. Even when the mask is not correctly worn, the system predicts "with mask" for the RFMD trained model throughout run time. In contrast, the system predicts that there will be no mask during RTFMD training, when face covers only mouth. Furthermore, the proposed approach and datasets were compared with existing approaches and datasets in terms of accuracy.

The future work on face mask detection application is to detect the other categories of face masks such as 3d masks, transparent masks and human appearance masks.

REFERENCES

[1] Worldometer, COVID-19 Coronavirus Pandemic, https://www.worldometers.info/corona-virus/, 2020, Online, accessed on June 10, 2020.

[2] WHO, Coronavirus disease (COVID-19) Situation Report, https://www.who.int/emergencies/diseases/novel-coronavirus-2019/situation-reports/, 2020, Online, accessed on June 10, 2020.

[3] World Health Organization, Coronavirus disease (COVID-19) advice for the public, https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public, Online, accessed June 10, 2020.

[4] Jiang, M., Fan, X., Yan, H. (2020). Retinamask: A face mask detector. arXiv preprint arXiv:2005.03950. https://arxiv.org/abs/2005.03950.

[5] Wang, Z., Wang, G., Huang, B., Xiong, Z., Hong, Q., Wu, H., Yi, P., Jiang, K., Wang, N., Pei, Y., Chen, H., Miao, Y., Huang, Z., Liang, J. (2020). Masked face recognition dataset and application. arXiv preprint arXiv:2003.09093. https://arxiv.org/abs/2003.09093.

[6] Ge, S., Li, J., Ye, Q., Luo, Z. (2017). Detecting masked faces in the wild with LLE-CNNs. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 426-434. https://doi.org/10.1109/CVPR.2017.53

[7] Deng, J., Guo, J., Zhou, Y., Yu, J., Kotsia, I., Zafeiriou, S. (2019). Retinaface: Single-stage dense face localisation in the wild. arXiv preprint arXiv:1905.00641. https://arxiv.org/abs/1905.00641.

[8] Zhang, F., Fan, X., Ai, G., Song, J., Qin, Y., Wu, J. (2019). Accurate face detection for high performance. arXiv preprint arXiv:1905.01585. https://arxiv.org/abs/1905.01585.

[9] Kumar Rai, R., Gour, P., Singh, B. (2012). Underwater image segmentation using CLAHE enhancement and thresholding. International Journal of Emerging Technology and Advanced Engineering, 2(1): 118-123.

[10] Yang, X.Y., Liang, N.N., Zhou, W., Lu, H.M. (2020). A face detection method based on skin color model and improved AdaBoost algorithm. Traitement du Signal, 37(6): 929-937. https://doi.org/10.18280/ts.370606

[11] Wang, Y.N., Yang, Y.M., Zhang, P.Y. (2020). Gesture feature extraction and recognition based on image processing. Traitement du Signal, 37(5): 873-880. https://doi.org/10.18280/ts.370521

[12] Simonyan, K., Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. https://arxiv.org/abs/1409.1556.

[13] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818-2826. https://doi.org/10.1109/CVPR.2016.308

[14] Guo, Y., Shi, H., Kumar, A., Grauman, K., Rosing, T., Feris, R. (2019). SpotTune: Transfer learning through adaptive fine-tuning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4805-4814. https://doi.org/10.1109/CVPR.2019.00494

[15] Dalgleish, T., Williams, J.M.G., Golden, A.M.J., Perkins, N., Barrett, L.F., Barnard, P.J., Yeung, C.A., Murphy, V., Elward, R., Tchanturia, K., Watkins, E. (2007). Reduced specificity of autobiographical memory and depression: the role of executive control. Journal of Experimental Psychology: General, 136(1): 23-42. http://dx.doi.org/10.1037/0096-3445.136.1.23

[16] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4510-4520. https://doi.org/10.1109/CVPR.2018.00474

[17] Duan, Y., Zhou, L., Wu, Y. (2016). Facial expression recognition based on convolution neural network. In 2nd International Conference on Computer Engineering, Information Science & Application Technology (ICCIA 2017), pp. 4077-4081.

[18] Chollet, F. (2017). Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1251-1258. https://arxiv.org/abs/1610.02357.

[19] Jayaswal, R., Dixit, M. (2021). A framework for anomaly classification using deep transfer learning approach. Revue d'Intelligence Artificielle, 35(3): 255-263. https://doi.org/10.18280/ria.350309

[20] Sajja, T.K., Devarapalli, R.M., Kalluri, H.K. (2019). Lung cancer detection based on CT scan images by using deep transfer learning. Traitement du Signal, 36(4): 339-344. https://doi.org/10.18280/ts.360406

[21] Bazi, Y., Al Rahhal, M.M., Alhichri, H., Alajlan, N. (2019). Simple yet effective fine-tuning of deep CNNs using an auxiliary classification loss for remote sensing scene classification. Remote Sensing, 11(24): 2908. https://doi.org/10.3390/rs11242908

[22] Ruder, S. (2016). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747. https://arxiv.org/abs/1609.04747.

[23] Duchi, J.C., Bartlett, P.L., Wainwright, M.J. (2012). Randomized smoothing for (parallel) stochastic optimization. In 2012 IEEE 51st IEEE Conference on Decision and Control (CDC), Maui, HI, USA, pp. 5442-5444. https://doi.org/10.1109/CDC.2012.6426698

[24] Kingma, D.P., Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. https://arxiv.org/abs/1412.6980.

[25] Zeiler, M.D. (2012). Adadelta: An adaptive learning rate method. arXiv preprint arXiv:1212.5701. https://arxiv.org/abs/1212.5701.

[26] Sutskever, I., Martens, J., Dahl, G., Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In International Conference on Machine Learning, pp. 1139-1147.

[27] McMahan, H.B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D., Chikkerur, S., Liu, D., Wattenberg, M., Hrafnkelsson, A.M., Boulos, T., Kubica, J. (2013). Ad click prediction: A view from the trenches. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1222-1230. https://doi.org/10.1145/2487575.2488200

[28] Probst, P., Boulesteix, A.L., Bischl, B. (2019). Tunability: importance of hyperparameters of machine learning algorithms. The Journal of Machine Learning Research, 20(1): 1934-1965. http://jmlr.org/papers/v20/18-444.html.

[29] Paul, S., Kurin, V., Whiteson, S. (2019). Fast efficient hyperparameter tuning for policy gradients. arXiv preprint arXiv:1902.06583. https://arxiv.org/abs/1902.06583.

[30] Jayaswal, R., Jha, J. (2017). A hybrid approach for image retrieval using visual descriptors. In 2017 International Conference on Computing, Communication and Automation (ICCCA), Greater Noida, India, pp. 1125-1130. https://doi.org/10.1109/CCAA.2017.8229965

[31] Jayaswal, R., Dixit, M. (2020). Comparative analysis of human face recognition by traditional methods and deep learning in real-time environment. In 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT), Gwalior, India, pp. 66-71. https://doi.org/10.1109/CSNT48778.2020.9115779

[32] Nagrath, P., Jain, R., Madan, A., Arora, R., Kataria, P., Hemanth, J. (2021). SSDMNV2: A real time DNN-based face mask detection system using single shot multibox detector and MobileNetV2. Sustainable Cities and Society, 66: 102692. https://doi.org/10.1016/j.scs.2020.102692

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