Indonesian Journal of Electrical Engineering and Computer Science
Vol. 29, No. 1, January 2023, pp. 304~314
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v29.i1.pp304-314

Increasing validation accuracy of a face mask detection by new deep learning model-based classification

Mohanad Azeez Joodi, Muna Hadi Saleh, Dheyaa Jasim Kadhim
Department of Electrical Engineering, College of Engineering, University of Baghdad, Baghdad, Iraq

Article Info

Article history:

Received May 29, 2022
Revised Sep 12, 2022
Accepted Sep 29, 2022

Keywords:

Deep learning CNN model
Face mask detection
Features vector
Haar cascade model
Validation accuracy

ABSTRACT

During COVID-19, wearing a mask was globally mandated in various workplaces, departments, and offices. New deep learning convolutional neural network (CNN) based classifications were proposed to increase the validation accuracy of face mask detection. This work introduces a face mask model that is able to recognize whether a person is wearing mask or not. The proposed model has two stages to detect and recognize the face mask; at the first stage, the Haar cascade detector is used to detect the face, while at the second stage, the proposed CNN model is used as a classification model that is built from scratch. The experiment was applied on masked faces (MAFA) dataset with images of 160x160 pixels size and RGB color. The model achieved lower computational complexity and number of layers, while being more reliable compared with other algorithms applied to recognize face masks. The findings reveal that the model's validation accuracy reaches 97.55% to 98.43% at different learning rates and different values of features vector in the dense layer, which represents a neural network layer that is connected deeply of the CNN proposed model training. Finally, the suggested model enhances recognition performance parameters such as precision, recall, and area under the curve (AUC).

Corresponding Author:

Mohanad Azeez Joodi
Department of Electrical Engineering, College of Engineering, University of Baghdad
Jadarayah, Baghdad, Iraq
Email: m.mohammad1702d@coeng.uobaghdad.edu.iq

1. INTRODUCTION

The corona virus disease of 2019 has wreaked havoc around the globe. Wearing face masks in public areas is a crucial protective step for people. To prevent the infection from spreading, certain districts have made wearing a mask mandatory in public locations. Only a few studies have looked at using image analysis to recognize face masks automatically. A light weight, deep learning-based face mask detector was presented for embedded systems with low computation needs and good productivity [1].

Face detection is a visual technique for detecting human faces in photographs. This strategy is critical to the success of the classification of human faces. Furthermore, real-time work on low-cost devices is required for practical applications of face recognition. It has been subjected to a variety of traditional procedures, but the accuracy is limited. The success of the deep learning method for extracting features from objects, on the other hand, has encouraged its employment to identify faces and backgrounds [2].

To better recognize small faces, a new scale-invariant face detector, known as the smallest face attention (SFA) face detector has been developed. In principle, it is a multi-branch face detection system that prioritizes small-scale faces. Large-scale options can be combined with feature maps from surrounding branches to make it easier to detect small-scale hard faces. Ultimately, multistage coaching and testing were used at the same time to make this model robust at many scales.

SFA significantly enhances face identification performance, especially on small faces, according to extensive testing. On challenging face detection benchmarks, as well as the WIDER FACE and face detection benchmark (FDDB) datasets, this technique achieves good detection performance with competitive speed [3], [4]. Face detection is greatly improved by convolutional neural networks with a lot of training. However, due to basic problems such as high computational cost and lengthy calculation, current convolutional neural network (CNN) based face detectors are unsuitable for many applications.

A real-time solution to face detection was created using a single end-to-end deep neural network with multi-scale feature maps, multi-earlier aspect ratios, and confidence correction. The multi-scale feature maps alleviate the difficulty of detecting small faces, while the aforementioned multi-scale aspect ratios reduce processing costs and trust rectification, which is in line with biological intuition and might further improve the detection rate. The suggested method, FDDB, outperformed the latest generation of CNN based algorithms in a public benchmark, but with fewer limitations [5], [6]. Face detection is critical in the development of facial recognition, expression, tracking, and classification.

Traditional techniques have difficulties in a variety of tough scenarios, such as non-frontal faces, occlusions, and intricate backdrops. Convolutional neural network (CNN) techniques generate remarkable results, despite a large number of calculations. As a result, CNN is incompatible with low-cost CPUs and requires high-end hardware.

This study develops a light architecture for real-time face identification using CNN. The proposed architecture consists of two main modules: a backbone for extracting facial traits and a multi layer survey for making predictions on several scales. Photos with a video graphics array (VGA) resolution are also applicable [7], [8].

As part of a powerful rotation-invariant multi-view face detector, the width-first-search (WFS) tree detector structure, the Vector Boost algorithm for learning strong classifications with vector output, the weak learning method based on domain partitioning, some functions in granular space, and the heuristic search for sparse function selection have all been proposed. As a result, the multi-view face detector achieves minimal computational complexity, wide detection range, and high detection accuracy on both traditional test sets and real-life photographs [9]. Table 1 (see Appendix) shows some related works with its contribution in the field of face mask detection and recognitions. The main contribution of this work is introducing a face mask model that is able to recognize whether a person is wearing mask or not. The proposed work has two stages to detect and recognize the face mask; at the first stage, the Haar cascade detector is used to detect the face, while at the second stage, the proposed CNN model is used as a classification model that is built from scratch. The model achieved lower computational complexity and number of layers. The proposed model improves the performance metrics of recognition in terms of precision, recall, and area under curve (AUC) in comparison with other methods that use different algorithms with the masked faces (MAFA) dataset is a masked face detection benchmark dataset, of which images are collected from Internet images. MAFA contains 30,811 images and 35,806 masked faces.

The organization of this research as follows: Section 1 describes the research background about mask face which is a critical strategy for minimizing COVID-19 transmission and saving lives during the COVID-19 outbreak. Face recognition systems and some literature survey about face recognition techniques that adopted upgraded convolutional neural networks. Section 2 presents the proposed model architecture of face mask detection consists of two stages: face detection and classification (with and without mask). Section 3 demonstrates the experimental results and discussion consisting of three subsections: the first section presents the MAFA faces datasets used to prepare the data, which contain the images of masked and unmasked faces; the second section is training the MAFA data set, and the third section contains classification performance in comparison with other methods that use different algorithms with the MAFA FACES dataset. Section 4 discusses the main contributions and conclusions that come with this work.

2. METHOD

The proposed model for face mask detection consists of two stages: face detection and classification (with and without mask), as shown in Figure 1. The face detection step is performed to detect the face region. The Viola-Jones Haar cascade detector is used to detect the facial region [10]. This detector performs feature extraction by Haar cascade detector with 160x160 window size. The detected face regions are cut and then input to the CNN proposed model classification. A less complex and more reliable built-from-scratch model are introduced and apply it on the MAFA dataset images to recognize the regions of masked and unmasked faces. The proposed model and its architecture are shown in Table 2 and Table 3. It consists of convolutional, batch normalization, activation, max pooling, and fully connected layers (which represents a neural network layer that is connected deeply of the CNN proposed model training) [11].

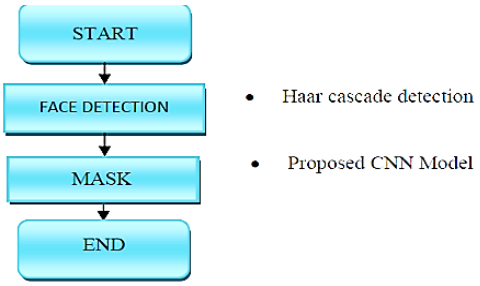

Figure 1. Procedure of proposed model

Table 2. The CNN proposed model

Layer (Type)	Output (shape)	Parameters
conv2d (Conv2D)	(None, 158, 158, 32)	896
conv2d_1 (Conv2D)	(None, 154, 154, 32)	25632
conv2d_2 (Conv2D)	(None, 150, 150, 64)	51264
batch normalization	Batch No (None, 150, 150, 64)	256
activation (Activation)	(None, 150, 150, 64)	0
max_pooling2d (MaxPooling2D)	(None, 75, 75, 64)	0
conv2d_3 (Conv2D)	(None, 73, 73, 64)	36928
batch_normalization_1	(None, 73, 73, 64)	256
activation_1 (Activation)	(None, 73, 73, 64)	0
max_pooling2d_1 (MaxPooling2)	(None, 36, 36, 64)	0
conv2d_4 (Conv2D)	(None, 34, 34, 64)	36928
batch_normalization_2 (Batch)	(None, 34, 34, 64)	256
activation_2 (Activation)	(None, 34, 34, 64)	0
max_pooling2d_2 (MaxPooling2)	(None, 17, 17, 64)	0
flatten (Flatten)	(None, 18496)	0
dense (Dense)	(None, 250)	4624250
activation_3 (Activation)	(None, 250)	0
dropout (Dropout)	(None, 250)	0
dense_1 (Dense)	(None, 1)	251
activation_4 (Activation)	(None, 1)	0

Table 3. Architecture of the proposed CNN model layers

Layer	Kernels size, Stride	Output image shape
Input image	-	(158,158,3)
Conv 1	32	(158,158)
Conv 2	32	(154,154)
Conv 3	64	(150,150)
Batch normalization	64	(150,150)
RELU	-	(150,150)
Max.pooling 1	2*2,2	(75,75)
Conv 4	64	(73,73)
Batch normalization 1	64	(73,73)
RELU	-	(73,73)
Max.pooling 2	2*2,2	(36,36)
Conv 5	64	(34,34)
Batch normalization 2	64	(34,34)
RELU	-	(34,34)
Max pooling 3	2*2,2	(17,17)
Flatten	-	18496 vectors
Dense	-	250 vectors
RELU	-	250 units
Dropout	0.5	250 units
Dens1	-	250 units
Sigmoid	-	1 unit

The process of convolution refers to the convolution of the two main functions (f) and (g), formulated as ($f * g$), whereas the output is (n). The (1) defines the expression for a one-dimensional convolution.

$$(f * g)(n) = \sum_m f(m)g(n-m) \quad (1)$$

Convolution can also be used in two-dimensional (2D) digital images, where *A* is a 2D image with (*i* * *j*) dimensions, *K* is a filter with (*m* * *n*) dimensions, and *F* is the feature map. The filter *K* is convolved with image *A* to produce the output *F*. In this case, the process shall be defined by (2) because it is commutative, and the 2D equation can be written as given by (3).

$$F(i,j) = (A * K)(i,j) = \sum_m \sum_n A(m,n)K(i-m, j-n) \quad (2)$$

$$F(i,j) = (A * K)(i,j) = \sum_m \sum_n A(i-m, j-n)K(m,n) \quad (3)$$

The RELU activation function is used to neglect the negative values. The adaptive moment estimation (Adam) is an optimization algorithm that can be used instead of the classical SGD procedure to update network parameters based on training data [10].

Adam algorithm has been shown empirically to demonstrate that the model can converge faster than the other algorithms. Adam simply combines the functions of the RMSprop and the Stochastic Gradient Descent with momentum. In this algorithm, the squared gradients are utilized to weigh the learning rate, as in the RMSprop, while benefiting from the momentum through the utilization of the gradient's moving average rather than the gradient itself, as in the SGD. Adam is an adaptive approach for learning rate; in other words, it performs the computation of each single learning rate for various metrics. Adam (coined from adaptive moment estimation) is termed so due to the fact that it utilizes the 1st and 2nd gradient moments to achieve adaptation to the learning rate for each neural network's weight [12]. N-th moment of a random variable is defined as the expected value of that variable to the power of n, as in (4).

$$m_n = E[X^n] \quad (4)$$

Where *m* refers to the moment while *X* refers to the random variable. As the evaluation of the gradient of the cost function of neural network is performed on a random small dataset, it is possible to describe it as a random variable. The 1st and the 2nd moments refer to the mean and the uncentered variance, respectively. To achieve moments estimation, Adam employs exponentially moving averages, the computation of which being achieved on the gradient that is evaluated on a current mini-batch, as expressed as in (5) and (6).

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \quad (5)$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) {g_t}^2 \quad (6)$$

Where *m* and *v* refer to moving averages, g refers to the gradient on current mini-batch, and betas refer to the newly introduced hyper-parameters of the algorithm. The good default values of *m* and *v* are 0.9 and 0.999 respectively. The good default values of *m* and *v* are 0.9 and 0.999 respectively. The initialization of the vectors that belong to the moving averages occurs with zeros at the first iteration, *m* and *v* represent the estimates of 1st and 2nd moments as in (7) and (8).

$$E[m_t] = E[g_t] \quad (7)$$

$$E[v_t] = E[{g_t}^2] \quad (8)$$

The expected values of the estimators must be equal to the parameter that needs to be estimated. In these calculations, it happens that the parameter and the expected value are the same. In the case that these properties are correct, this would imply that the estimators are not biased. The lowest first values of the gradients make contributions to the collective value, since they become subjected to multiplications by smaller and smaller beta values. The equation for moving average can be given as in (9).

$$m_t = (1-\beta_1) \sum_{i=0}^{t} \beta_1^{t-i} g_i \quad (9)$$

At this point, by having a look at the expected value of m, the relation to the true first moment can be observed, and then the corrections for the discrepancy will be able to make it as in (10), (11), and (12).

$$E[m_t] = E\left[(1-\beta_1) \sum_{i=1}^{t} \beta_1^{t-i} g_i\right] \quad (10)$$

$$E[m_t] = E[g_i](1-\beta_1) \sum_{i=0}^{t} \beta_1^{t-i} + \zeta \quad (11)$$

$$E[m_t] = E[g_i]\left(1-{\beta_1}^t\right) + \zeta \quad (12)$$

After thought, the bias correction step should be made, since the estimator needs to be corrected in a way that the expected value is wanted. As in (13) and (14) present the final expressions of the estimator of the model.

$$m_t^{\wedge} = \frac{m_t}{1-\beta_1^t} \tag{13}$$

$$v_t^{\wedge} = \frac{v_t}{1-\beta_2^t} \tag{14}$$

The final step is to utilize these moving averages in weighing each metric's learning rate individually. By using Adam algorithm, this can be achieved very easily, where weight updates can be obtained through the application as in (15).

$$w_t = w_{t-1} - \zeta \frac{m_t^{\wedge}}{\sqrt{v_t^{\wedge}+\epsilon}} \tag{15}$$

Where w refers to model weights which is the update rule for Adam, and ζ represents the Adam learning rate.

Dropout refers to a commonly used generalization technique. During each training period, neurons are released randomly. In this way, the power of function selection is evenly distributed across the group of neurons, while forcing the model to learn multiple independent functions. During the training process, the fallen neuron will not be part of the backward or the forward propagation to avoid over fitting. Instead, the large-scale network is used to make predictions during the testing process. The predicting process is configured at the first layer 158x158x32 and the output is a feature map which has the size 17x17x64. Binary Cross-Entropy has been used as a loss function; it gives a measure in the interval 0-1 of how separated two values are. Binary cross-entropy can be defined as the loss function given as in (16) [13].

$$loss(\alpha, \acute{\alpha}) = -\big(alog(\acute{\alpha}) + (1-a)log(1-\acute{\alpha})\big) \tag{16}$$

Where ά is the value returned by the model and α is the true label value. It is common to minimize (α, ά) for multiple images at the same time.

3. RESULTS AND DISCUSSION

The result part of this work consists of three sections to achieve the CNN proposed model classification. The first section presents the datasets used to prepare the data, which contain the images of masked and unmasked faces. The second section describes the training data and the selection of the different learning rates and different features vectors. The third section explains the use of some performance measures to determine the quality and validation accuracy of the model.

3.1. Data set

For the experiments, the dataset MAFA was used as in [14], which consists of 35,806 masked faces with a minimum size of 160x160. From this MAFA dataset, 5902 images were selected which contain frontal faces. The dataset was divided into two parts for training and validation sets, with 4759 and 1143 images, respectively. Figures 2-3 can show example images in the MAFA dataset.

Figure 2. Faces with masks in MAFA dataset

Figure 3. Faces without mask in MAFA dataset

3.2. Training

The training procedure included two stages. The first stage used 0.001 Adam optimizer learning rate value in the initial stage and calculated the entire performance metrics. The second stage changed the learning rate to 0.0001, but also following the same procedure. The binary cross-entropy loss between the training and validation was measured at different learning rates that indicate the performance of the proposed model, using batch size value of 32. The training process took place in 30 Epoch; the selected features vector values were 250. Figure 4 shows the binary cross-entropy loss between training and validation datasets at a 0.001 learning rate. The results showed a loss of 0.0727, which is lower than that achieved at a learning rate of 0.0001 (0.1094). Figure 5 illustrates the accuracy between the training and validation data. Figures 6-7 illustrate the validation accuracy and loss between the training and validation datasets, with Epoch at features vector of 1500 values selected as input to the dense layer.

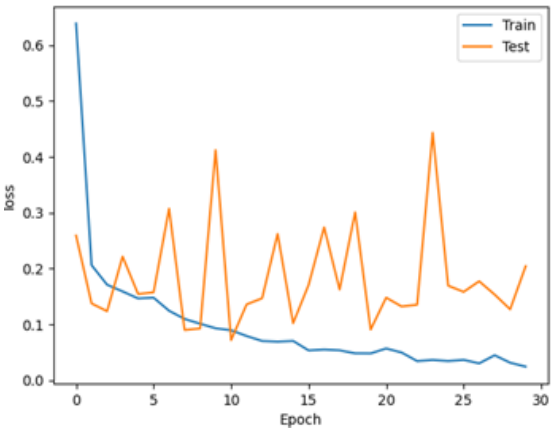

Figure 4. The training and validation loss at features vector of 250 values

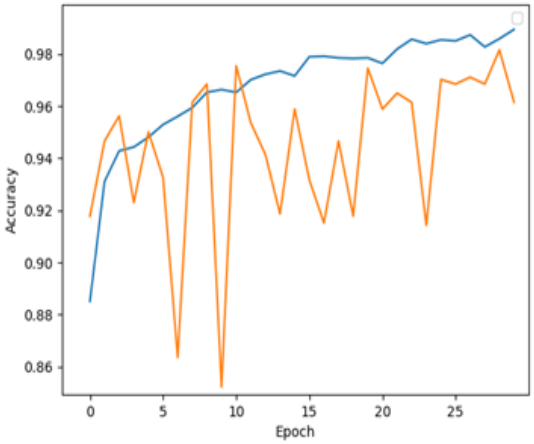

Figure 5. The accuracy of the training and validation at features vector of 250 values

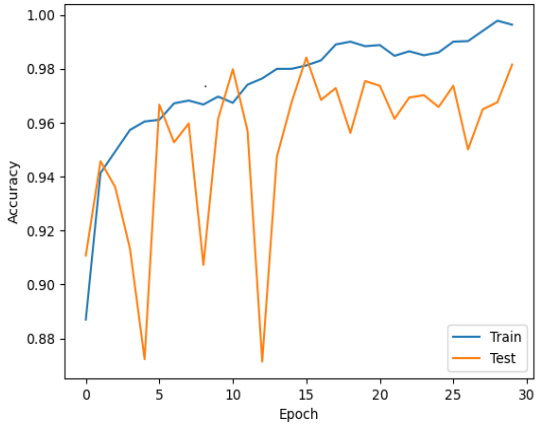

Figure 6. The accuracy of the training and validation a features vector of 1500 values

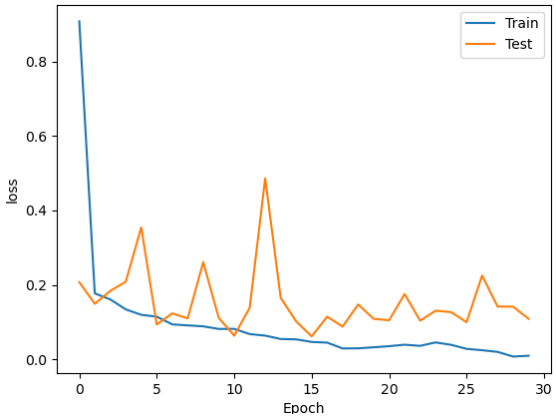

Figure 7. The training and validation loss at features vector of 1500 values

3.3. Performance

The performance of the proposed model was evaluated using five metrics, which are the precision or Specificity (is a measure of the fraction of negative class that is correctly classified), recall (true positiverate is a measure of the fraction of positive class that is correctly classified), validation accuracy (is the number of correct predictions divided by the total number of instances evaluated) , training error (the number of incorrect predictions divided by the total number of instances evaluated), and false positive rate (FPR), as given as in (17), (18), (19), (20), (21), and (22). The precision and area under the curve (AUC) metrics indicate the model's accuracy and classification quality as determined by the classification results. The retrieval metric that presents

the ability to find all of the relevant objects in a dataset is given in Table 4 at different learning rates, the loss between training and validation datasets at a 0.001 learning rate, the results showed a loss of 0.0727, the validation accuracy reached to 97.55% which was lower than that achieved at a learning rate of 0.0001 (0.1094), although the validation accuracy reached to 98.34%**,** while 250 features vector values were selected for both.

$$Precision = \frac{TP}{TP + FP} \tag{17}$$

$$TPR\ (True\ positive\ rate) = Recall = \frac{TP}{TP + FN} \tag{18}$$

$$Validation\ Acc = \frac{(TP+TN)}{(TP + FP + TN+FN)} \tag{19}$$

$$Training\ error = \frac{(FP + FN)}{(TP + FP + TN + FN)} \tag{20}$$

$$Specificity = \frac{TN}{TN+FP} \tag{21}$$

$$FPR\ (False\ positive\ rate) = 1 - Specificity = \frac{FP}{TN+FP} \tag{22}$$

Where true positive (TP): the class is positive and the model prediction also positive. True negative (TN): the class is negative and the model prediction also negative. False positive (FP): the model was misclassified in the negative class. False negative (FN): the model was misclassified in the positive class. Where real positive class (P)=TP+FN, real negative class (N)=FP+TN. Different features vector values were selected in the dense layer used in the training dataset; the validation accuracy reached to 98.43% at features vector value of 1500, whereas the loss value was 0.0623 and the mean square error value was 0.0132 and calculates all the metrics performance (confusion matrix) as shown in Table 5.

Table 4. The confusion matrix at 30 epochs for different Adam optimizer learning rates

Learning rate	True positive	True negative	False positive	False negative	Precision	Recall (TPR)	FPR	AUC	Validation _accuracy	Loss	Mean square error
0.001 (best test)	583	532	15	13	0.9749	0.9782	0.0274	0.9962	0.9755	0.0727	0.0197
0.0001	585	539	8	11	0.9865	0.9815	0.0146	0.9928	0.9834	0.1094	0.0155

Table 5. The validation accuracy and loss with different features vector values

Features vector	True positive	True negative	False positive	False negative	Precision	Recall (TPR)	FPR	AUC	Validation _accuracy	Loss	Mean square error
100	582	533	14	14	0.9765	0.9765	0.02559	0.9969	97.55%	0.0769	0.0200
250	583	532	15	13	0.9749	0.9782	0.02742	0.9962	97.55%	0.0727	0.0197
500	588	525	22	8	0.9639	0.9866	0.04021	0.9951	97.38%	0.0804	0.0214
900	589	529	18	7	0.9703	0.9883	0.03290	0.9957	97.81%	0.0687	0.0169
1200	585	530	17	11	0.9718	0.9815	0.03107	0.9952	97.55%	0.0807	0.0209
1500 (best test)	588	537	10	8	0.9833	0.9866	0.01828	0.9966	98.43%	0.0623	0.0132
1600	584	530	17	12	0.9717	0.9799	0.03107	0.9963	97.46%	0.0708	0.0188
1700	587	529	18	9	0.9702	0.9849	0.03290	0.9959	97.46%	0.0695	0.0181
2000	580	535	12	16	0.9797	0.9732	0.02193	0.9948	97.55%	0.0801	0.0193
3000	582	518	29	14	0.9525	0.9765	0.05301	0.9947	96.24%	0.0967	0.0276

Figures 8-9 show charts representing different features vector values selected in the CNN flatten layer in the training dataset and their effects on validation accuracy and loss. This change in the features vector describes the changes in accuracy and loss, which will therefore affect the quality and accuracy of the classification model.

The accuracy of the proposed model can reach up to 97.55% when used to detect objects in 160x160 pixels images with features vector of 250 values, which is higher than that achieved by other methods. Also,

the values of the performance metrics are higher, and the model showed the ability to perform faster. Also, the accuracy reached up to 98.43% when 1500 values of features vector were selected in the dense layer of our proposed model.

Table 6 illustrates the values of the performance metrics of precision and recall of the proposed model in comparison with other methods that use different algorithms with the MAFA FACES dataset. The accuracy of the proposed model can reach up to 97.55% when used to detect objects in 160x160 images with features vector of 250 values, which is higher than that achieved by other methods. Also, the values of the performance metrics are higher, and the model showed the ability to perform faster. Also, the accuracy reached up to 98.43% when 1500 values of features vector were selected in the dense layer of our proposed model.

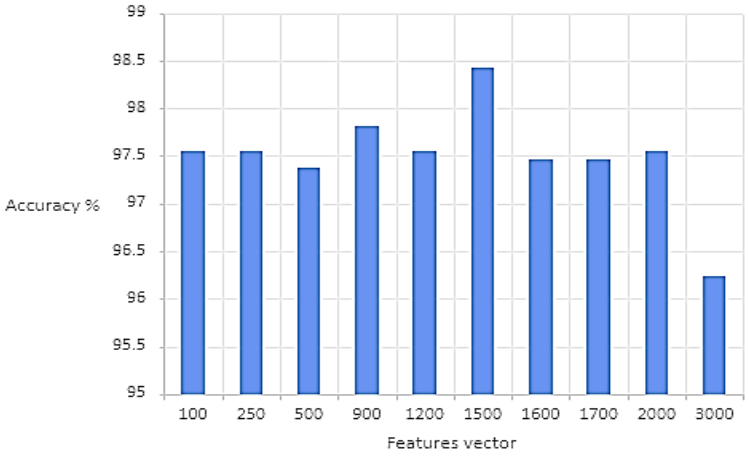

Figure 8. The validation accuracy versus features vector

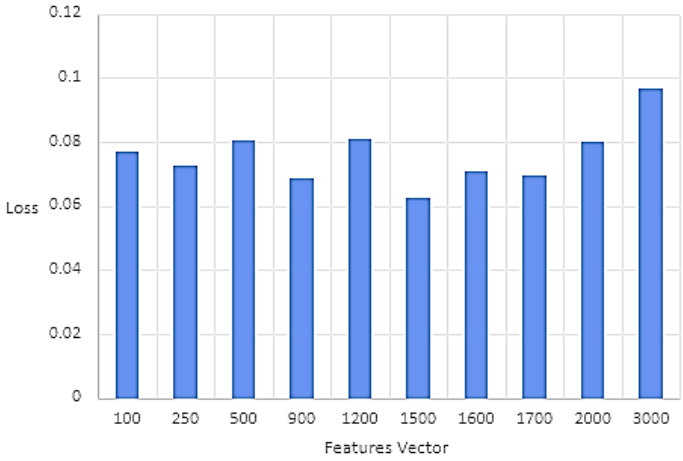

Figure 9. The loss versus features vector

Table 6. Comparison of the metrics performance between the proposed model and other algorithms

Reference no.	Year	Precision	Recall	Accuracy
[14]	2017	0.828	0.89	89%
[15]	2020	0.83	0.901	90.1%
Our proposed model of 250 values features vector is chosen	2022	0.9749	0.9782	97.55%
Our proposed model of 1500 values features vector is chosen	2022	0.9833	0.9866	98.43%

4. CONCLUSION

This work presented a Haar cascade detector within a proposed built-from-scratch CNN image classification model. The model showed lower computational complexity and higher reliability with the need

for fewer layers to treat RGB images with size of 160x160 pixels, when compared with other algorithms. The proposed model improved the performance metrics of validation accuracy, precision, recall and AUC when applied on the images of the MAFA data set, with the application of different values of Adam optimizer learning rate and features vector. Another contribution of this work was the use of different values of features vector thereby influencing the performance metrics of classification, as demonstrated by the reduction of the loss between the validation and training datasets. This system can be applied practically in any crowded building when it is necessary to recognize masked from unmasked faces.

APPENDIX

Table 1. Contributions from related works that used certain techniques to detect and recognize face masks

Ref. No.	Year	Method and Techniques	Contribution	Dataset used
[14]	2017	To extract the candidate face regions from the input image and show them with large descriptors, the proposal module first concatenated two pertained CNNs. Using the local linear integration (LLE) algorithm and dictionaries, the integration module was used to convert these descriptors into a match-based descriptor. Faceless faces, masked faces, and synthesized regular faces were all used in the training.	Precision 82.8% Recall 89%	MAFA FACE
[15]	2020	In this system's real-time mask detection and recognition technique, the Haar cascade classifier was used to detect the face, and the YOLOv3 algorithm was utilized to detect the mask and determine whether the individual was wearing it or not.	Acc 90.1%	MAFA FACE
[16]	2020	On the basis of this analysis, Advanced Multitasking Cascaded Convolutional Neural Networks (MTCNN) were used to detect the mask-induced area of occlusion, find the LBP (Local Binary Pattern) function of the non-mask-induced area of occlusion hidden, and finally enter the function into the vector machine face recognition support.	Acc 82.6%	WIDER FACE + MAFA FACE
[17]	2020	A very fast image pre-processing with the mask in the center over the faces are proposed. Classification and detection of a masked person, features extraction and CNN are used.	Acc 96.37%	GitHUB
[1]	2021	A light weight single-face mask detector based on deep learning was proposed to meet the minimal computing requirements of embedded devices while providing great productivity. Two new strategies to improve the model feature extraction process were proposed. First, a new residual contextual attention module was presented to extract information from the rich surroundings and focus on the critical regions linked with the face mask. Second, a unique assistive task based on synthesized Gaussian heat map regression was proposed to learn additional discriminatory features for masked and unmasked faces.	Acc 93.8%	WIDER FACE + MAFA FACE
[18]	2021	A new data collection and two alternative approaches were provided to distinguish masked and unmasked faces in real-time. The first technique employed an object detection model. The second technique used a YOLO face detector to detect faces (whether masked or not), followed by a unique, fast, yet effective CNN architecture to categorize the faces into masked and unmasked categories.	Acc 97% Precision 92% Recall 97%	WIDER FACE + MAFA FACE
[19]	2021	In an automated face mask identification system, median filtering and the back propagation neural network (BPNN) were proposed.	Acc 91% Sensitivity 90% Specialist 92%	Small dataset from Internet
[20]	2021	The first high-performance, single-stage face mask detector, Retina Facemask was proposed. To begin, annotations were used to construct a new dataset to address the problem that previous studies were unable to distinguish between right and erroneous mask-wearing situations. Second, a contextual attention module was suggested that would focus on learning the distinguishable traits associated with different face mask usage states.	Acc 94.8%	MAFA FACE + FMD FACE
[21]	2021	A new mask identification and classification technique that includes both transfer learning and deep learning was proposed. A replacement technique for mask recognition that mixes transfer learning and economical-Yolov3 was suggested, with an efficient net because of the use of the backbone feature extraction network and CIoU as the loss function, to cut back the number of network parameters and improve mask detection accuracy.	Acc 96.03%	MAFA FACE + WIDER FACE
[22]	2021	A method was presented for obtaining partially rebuilt aspects of the obscured part of the face. The face was recognized using an existing deep learning approach. The occluded part of the face was first removed, and the remaining component of the face was subjected to Principal Component Analysis (PCA).	Acc 85% -95%	YALE FACE

Table 1. Contributions from related works that used certain techniques to detect and recognize face masks *(continue)*

Ref. No.	Year	Method and Techniques	Contribution	Dataset used
[23]	2021	Despite the fact that the initial dataset was fairly limited, the CNN model that applied the transfer learning technique showed exceptional accuracy. For the first time, a big face mask detection data set was utilized to train the model, while the much smaller initial face mask detection data set was used to modify and refine the previously developed model.	Acc 97.1%	Kaggle
[24]	2021	Two less difficult convolutions were presented. A Raspberry Pi 4 module was utilized to create Neural Network (CNN) based classifications that successfully identify data and recognize persons in real-time.	Acc 97.67%	Special data set only 1000 face masks
[25]	2021	To achieve short inference time and good accuracy, the suggested technique used a set of one- and two-stage detectors. ResNet50 was used as a starting point and the learning transfer idea was used to incorporate high-level semantic information into various function mappings.	Acc 98.2% Precision 98.92 Recall 98.24	MAFA + Special dataset.

REFERENCES

[1] X. Fan, M. Jiang, and H. Yan, "A deep learning based light-weight face mask detector with residual context attention and Gaussian heatmap to fight against COVID-19," *IEEE Access*, vol. 9, pp. 96964–96974, 2021, doi: 10.1109/access.2021.3095191.

[2] M. D. Putro and K.-H. Jo, "Fast face-CPU: a real-time fast face detector on CPU using deep learning," *2020 IEEE 29th International Symposium on Industrial Electronics (ISIE)*. IEEE, 2020, doi: 10.1109/isie45063.2020.9152400.

[3] S. Luo, X. Li, R. Zhu, and X. Zhang, "SFA: small faces attention face detector," *IEEE Access*, vol. 7, pp. 171609–171620, 2019, doi: 10.1109/access.2019.2955757.

[4] D. J. Kadhim and O. A. Mamad, "Improving IoT applications using a proposed routing protocol," *Journal of Engineering*, vol. 20, no. 11, 2014.

[5] C. Zheng, M. Yang, and C. Wang, "A real-time face detector based on an end-to-end CNN," *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2017, doi: 10.1109/iscid.2017.138.

[6] S. H. Abdulredah and D. J. Kadhim, "New approaches of cloud services access using tonido cloud server for real-time applications," *Journal of Engineering*, vol. 26, no. 8, pp. 83–99, 2020, doi: 10.31026/j.eng.2020.08.07.

[7] M. D. Putro, L. Kurnianggoro, and K.-H. Jo, "High performance and efficient real-time face detector on central processing unit based on convolutional neural network," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4449–4457, 2021, doi: 10.1109/tii.2020.3022501.

[8] S. Q. Jabbar and D. J. Kadhim, "A proposed adaptive bitrate scheme based on bandwidth prediction algorithm for smoothly video streaming," *Journal of Engineering*, vol. 27, no. 1, pp. 112–129, 2021, doi: 10.31026/10.31026/j.eng.2021.01.08.

[9] C. Huang, H. Ai, Y. Li, and S. Lao, "High-performance rotation invariant multiview face detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 4, pp. 671–686, 2007, doi: 10.1109/tpami.2007.1011.

[10] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, Dec. 2014, vol. 1, pp. I-511-I–518, doi: 10.48550/arXiv.1412.6980.

[11] M. A. Wani, F. A. Bhat, S. Afzal, and A. I. Khan, "Advances in deep learning," *Studies in Big Data*. Springer Singapore, 2020, doi: 10.1007/978-981-13-6794-6.

[12] V. Bushaev, "Adam — latest trends in deep learning optimization," *towardsdatascience.com*, 22 Oct. 2018, https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c. Accessed 28 March 2022.

[13] L. H. Hughes, M. Schmitt, L. Mou, Y. Wang, and X. X. Zhu, "Identifying corresponding patches in SAR and optical images with a pseudo-siamese CNN," *IEEE Geoscience and Remote Sensing Letters*, vol. 15, no. 5, pp. 784–788, 2018, doi: 10.1109/lgrs.2018.2799232.

[14] S. Ge, J. Li, Q. Ye, and Z. Luo, "Detecting masked faces in the wild with LLE-CNNs," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, doi: 10.1109/cvpr.2017.53.

[15] T. Q. Vinh and N. T. N. Anh, "Real-time face mask detector using YOLOv3 algorithm and Haar cascade classifier," *2020 International Conference on Advanced Computing and Applications (ACOMP)*. IEEE, 2020, doi: 10.1109/acomp50827.2020.00029.

[16] C. Qi and L. Yang, "Face recognition in the scene of wearing a mask," *2020 International Conference on Advance in Ambient Computing and Intelligence (ICAACI)*. IEEE, 2020, doi: 10.1109/icaaci50733.2020.00021.

[17] M. S. Islam, E. Haque Moon, M. A. Shaikat, and M. Jahangir Alam, "A novel approach to detect face mask using CNN," *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS)*. IEEE, 2020, doi: 10.1109/iciss49785.2020.9315927.

[18] S. Abbasi, H. Abdi, and A. Ahmadi, "A face-mask detection approach based on YOLO applied for a new collected dataset," *2021 26th International Computer Conference, Computer Society of Iran (CSICC)*. IEEE, 2021, doi: 10.1109/csicc52343.2021.9420599.

[19] Y. Ayyappa, P. Neelakanteswara, A. Bekkanti, Y. Tondeti, and C. Z. Basha, "Automatic face mask recognition system with FCM AND BPNN," *2021 5th International Conference on Computing Methodologies and Communication (ICCMC)*. IEEE, 2021, doi: 10.1109/iccmc51019.2021.9418243.

[20] X. Fan and M. Jiang, "RetinaFaceMask: A single stage face mask detector for assisting control of the COVID-19 pandemic," *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021, doi: 10.1109/smc52423.2021.9659271.

[21] X. Su, M. Gao, J. Ren, Y. Li, M. Dong, and X. Liu, "Face mask detection and classification via deep transfer learning," *Multimedia tools and applications*, vol. 81, no. 3, pp. 4475–4494, 2022, doi: 10.1007/s11042-021-11772-5.

[22] S. Malakar, W. Chiracharit, K. Chamnongthai, and T. Charoenpong, "Masked face recognition using principal component analysis and deep learning," *2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. IEEE, 2021, doi: 10.1109/ecti-con51831.2021.9454857.

[23] M. M. Boulos, "Facial recognition and face mask detection using machine learning techniques," Montclair State University, 2021.

[24] M. I. Amin, M. Adeel Hafeez, R. Touseef, and Q. Awais, "Person identification with masked face and thumb images under pandemic of COVID-19," *2021 7th International Conference on Control, Instrumentation and Automation (ICCIA)*. IEEE, 2021, doi: 10.1109/iccia52082.2021.9403577.
[25] S. Sethi, M. Kathuria, and T. Kaushik, "Face mask detection using deep learning: an approach to reduce risk of Coronavirus spread," *Journal of biomedical informatics*, vol. 120, p. 103848, Aug. 2021, doi: 10.1016/j.jbi.2021.103848.

BIOGRAPHIES OF AUTHORS

Mohanad Azeez Joodi received B.Sc. in Electrical Engineering from University of Baghdad (1991-1995), he received M.Sc. in Electrical Engineering/computer and control at Baghdad University (1999-2002) and now his PhD student in Electrical Engineering Department in Computer and control from University of Baghdad. He is currently a Faculty Member and Lecturer of the Electrical Engineering Department-College of Engineering-Baghdad University. He published many papers. His field of interest and his research works focus on AI, Soft Computing, Deep learning, Neural Network, Fuzzy Logic, optimization algorithms, Control theory, Computer Architecture. He can be contacted at email: m.mohammad1702d@coeng.uobaghdad.edu.iq.

Muna Hadi Saleh received B.Sc. in Control and Systems Engineering from University of Technology (1988-1989), she received M.Sc. in Control and Instrumentation Engineering from University of Technology (1995-1996) while my Ph.D. in Computer and Information Technology from University of Technology (2005-2006). Now I'm a Faculty Member of the Electrical Engineering Department-College of Engineering-Baghdad University. I'm published thirty-four papers, supervised M.Sc. and Ph.D. students. Finally, I'm researcher in many conferences, and I'm Scientific's' experience in IEEE, JESTEC, also I'm a membership in IAO. My field of interest and my research works focus on AI, Intelligent System, Soft Computing, Computer Control, Neural Network, Fuzzy Logic, Intelligent Methodology, Automation Control (Robot), Computer Architecture, Microprocessors, E-Learning, Data Mining, Genetic Algorithm, Rough Set. She can be contacted at email: dr.muna.h@coeng.uobaghdad.edu.iq.

Dheyaa Jasim Kadhim received B.Sc. in Electrical Engineering at Baghdad University; he received M.Sc. in Electrical Engineering at Baghdad University (1999-2002) and received Ph.D. in Communication and Information Engineering at Huazhong University of Science and Technology in China (2006-2009). He received PostDoc. in Information Technology at Wuhan University of Science and Technology in China (2011-2013). He is currently a Faculty Member and Associative Professor of the Electrical Engineering Department/Baghdad University, IEEE Member. During (2004-current), he joined Cisco Academy and Huawei Academy as an academic instructor and he get some important certificates from these academies. He supervised many master's and doctorate students. Finally, he published many papers and participated in many important international dheyaa@coeng.uobaghdad.edu.iq.