Available online at www.ijournalse.org

Emerging Science Journal

(ISSN: 2610-9182)

Vol. 7, No. 4, August, 2023

An Optimized Machine Learning and Deep Learning Framework for Facial and Masked Facial Recognition

Putthiporn Thanathamathee [1], Siriporn Sawangarreerak [2*], Prateep Kongkla [1], Dinna Nina Mohd Nizam [3]

[1] *School of Engineering and Technology, Walailak University, Nakhon Si Thammarat 80160, Thailand.*

[2] *School of Accountancy and Finance, Walailak University, Nakhon Si Thammarat 80160, Thailand.*

[3] *User Experience Research Group, Faculty of Computing and Informatics, Universiti Malaysia Sabah, 87000 W.P.Labuan, Malaysia.*

Abstract

In this study, we aimed to find an optimized approach to improving facial and masked facial recognition using machine learning and deep learning techniques. Prior studies only used a single machine learning model for classification and did not report optimal parameter values. In contrast, we utilized a grid search with hyperparameter tuning and nested cross-validation to achieve better results during the verification phase. We performed experiments on a large dataset of facial images with and without masks. Our findings showed that the SVM model with hyperparameter tuning had the highest accuracy compared to other models, achieving a recognition accuracy of 0.99912. The precision values for recognition without masks and with masks were 0.99925 and 0.98417, respectively. We tested our approach in real-life scenarios and found that it accurately identified masked individuals through facial recognition. Furthermore, our study stands out from others as it incorporates hyperparameter tuning and nested cross-validation during the verification phase to enhance the model's performance, generalization, and robustness while optimizing data utilization. Our optimized approach has potential implications for improving security systems in various domains, including public safety and healthcare.

Keywords:

Masked Face Recognition;
Deep Learning;
Hyperparameter Tuning;
Grid Search;
Nested Cross-Validation.

Article History:

Received:	14	February	2023
Revised:	13	May	2023
Accepted:	08	June	2023
Available online:	12	July	2023

1- Introduction

Due to the recent COVID-19 pandemic, many countries require individuals to wear masks in public spaces to prevent the spread of COVID-19. The use of traditional bio-metric-based techniques like fingerprint and facial recognition without masks has been prohibited due to the COVID pandemic since these techniques can lead to the transmission of the COVID-19 virus among users. Manual inspection of people wearing masks in public spaces is a challenging task [1]. As a result, an automated mask detection system must be built. Furthermore, wearing a mask also presents new challenges for traditional facial recognition applications, which are typically designed to reveal faces [2]. The existing facial recognition systems are not very accurate for masked faces [3].

The quick creation of trustworthy facial recognition technology that can recognize any person even when they are wearing a face mask is required due to this pandemic ailment [4]. Ejaz et al. [5] reduced the dimensionality and extracted features using Principal Component Analysis (PCA). They performed identity identification and estimated the average facial characteristics of each identity. While non-masked faces are recognized with an average accuracy of 95%, masked

* **CONTACT**: siriporn.sa@wu.ac.th

DOI: http://dx.doi.org/10.28991/ESJ-2023-07-04-010

faces are recognized with an average accuracy of 72%. Therefore, compared to non-masked faces, PCA shows poor identification rates for images of masked faces. A multi-threading technique was utilized by Maharani et al. [6] to construct the masked face recognition approach based on Haarcascade and MobileNet to detect masks and VGG16 and Triplet loss FaceNet to identify faces. With modifications to the backbone and loss function, Montero et al. [7] proposed a deep model based on ArcFace. They created a masked version of the original face recognition dataset via data augmentation, and then they tested ResNet-50 on the recognition of masked faces at a low computational cost. The mask-usage classification loss and ArcFace loss are then merged to create a new function called multi-task ArcFace (MTArcFace). The results of the experiments indicate that the classification of mask usage has an average accuracy of 99.78%.

Hariri [8] suggested a deep learning-based approach and a quantization-based strategy to cope with the recognition of the masked faces. Additionally, they employed the RMFRD dataset, which enables the selection of deep features from the collected regions using the three pre-trained deep VGG-16, AlexNet, and ResNet-50 models. The classification process is completed by using Multilayer Perceptron (MLP). According to experimental findings, the average recognition accuracy when masked is between 88.90% and 91.30%. The Embedding Un-masking Model (EUM), which is built on top of the current face recognition models, was proposed by Boutros et al. [9]. They also suggest a brand-new loss function known as the Self-restrained Triplet (SRT), which allowed the EUM to create embeddings that resemble those of unmasked faces belonging to the same identities. Golwalkar & Mehendale. [10] construct 128-d encodings to recognize masked faces using FaceMaskNet-21 and the deep metric learning technique. In order to achieve more active recognition of faces covered by a mask, they additionally incorporated HOG features. The performance received a testing accuracy score of 88.92%. WearMask3D is a 3D model-based method that Hong et al. [11] presented for enhancing face photos in various positions to their masked face counterparts. They have also shown that practicing with artificially generated 3D masks can increase the recognition accuracy of masked faces. Ejaz et al. [12] suggested utilizing a Multi-Task Cascaded Convolutional Neural Network (MTCNN) to detect the face areas.

The Google FaceNet embedding model is utilized for the extraction of facial features. Finally, Support Vector Machine has carried out the classification operation (SVM). They carried out two scenarios: the first involved using masked faces for testing and unmasked faces for training, while the second involved using both masked and unmasked faces for both training and testing. For both training and testing, the performance related to both masked and unmasked faces is 98.50%. Deng et al. [13] introduced the MFCosface masked face recognition algorithm, which is based on wide margin cosine loss and is optimized by including an attention-aware mechanism in the model, to recognize the important facial features. For masked face recognition, Li et al. [14] also presented an attention-based method and a cropping-based technique. The area around the eyes was highlighted in the attention-based portion using the Convolutional Block Attention Module (CBAM) with an accuracy of 92.61%. Based on the FaceNet model and the residual inception network of the Inception-ResNet-v1 architecture, Moungsouy et al. [15] developed a method for recognizing human faces in both mask- and non-mask-wearing situations. The simulated masks on the original face images are augmented for the training model. The features derived from the top half of face photos are then computed to demonstrate that the characteristics selected for face recognition are correct using heatmaps. The outcome demonstrates a remarkable accuracy of 99.2% in a scenario of faces wearing masks.

Talahua et al. [16] suggested a method for identifying people from images even when they are wearing a face mask. To recognize faces, a feedforward multilayer perceptron and feature extractor from the FaceNet model are utilized. According to the testing findings, there is an accuracy of 99.65% in determining whether or not a person is wearing a mask. For mask detection, mask type classification, mask position classification, and identity recognition, Song et al. [17] implemented the Spartan Face Detection and Facial Recognition System into practice. For mask detection, CNN, AlexNet, and VGG16 are used. Additionally, they applied SVM and XGBoost for classifiers, which have respective accuracy rates of 97.00% and 88.00%, respectively. Kim et al. [18] introduced a new approach called Adaface, which took image quality into account as a critical factor for enhancing face recognition accuracy. They had proposed a generalized loss function that utilized the feature paradigm to estimate image quality. This approach enabled a smooth transition between the ArcFace [19] and CosFace [20] methods, leading to improved accuracy for low-quality images while maintaining high accuracy for high-quality images.

Lu & Zhuang [21] generated a large number of facial images with masks using public face datasets to address the problem of training data shortage. They also proposed a new network architecture called the Upper-Lower Network (ULN) to efficiently recognize masked faces. The upper branch of ULN, which takes mask-free images as input, was pre-trained to provide supervisory information for the lower branch. They further divided the high-order semantic features into upper and lower parts, as the occlusion areas of masks usually appear in the lower parts of faces. The designed loss function forced the learned features of the lower branch to be similar to those of the upper branch with the same mask-free image inputs, but only the upper part of the features were similar to their mask counterparts. Ge et al. [22] proposed a Convolutional Visual Self-Attention Network (CVSAN) that incorporated self-attention to enhance the convolution operator. They achieved this by connecting a convolutional feature map, which enforced local features, to a self-attention feature map that could model long-range dependencies. To train the CVSAN model, they generated a

Masked VGGFace2 dataset using a face detection algorithm since there was no publicly available large-scale masked face data. Experimental results showed that the CVSAN algorithm significantly improved the performance of masked face recognition.

Wang et al. [23] proposed an improved version of Facenet for face recognition. They had utilized the ConvNeXt-T architecture as the backbone of their model and incorporated the ECA (Efficient Channel Attention) mechanism for efficient channel attention. This approach enhanced feature extraction for the visible parts of the face, resulting in more valuable information without increasing model complexity or reducing dimensionality. They had explored the effects of different attention mechanisms and dataset ratios on face mask recognition models and had constructed a large dataset of faces wearing masks for efficient model training. Experimental results showed that their model had achieved 99.76% accuracy for recognizing real faces wearing masks and a combined accuracy of 99.48% in challenging environments with extreme contrast and brightness. Table 1 presents an overview of the present models and their level of accuracy in recognizing masked faces.

Table 1. Provides a summary of the model and accuracy of masked face recognition systems

Method	Model	Accuracy	Real-world deployment
Ejaz et al. [5]	Viola-Jones algorithm + PCA	0.8350	No
Maharani et al. [6]	VGG16 + FaceNet	1.0000	Yes
Montero et al. [7]	MTArcFace	0.9978	No
Hariri [8]	VGG16 + BoF + MLP	0.9130	No
Golwalkar et al. [10]	FaceMaskNet-21	0.8892	Yes
Ejaz et al. [12]	MTCNN + FaceNet + SVM	0.9850	No
Moungsouy et al. [15]	MTCNN + Inception-ResNet-v1	0.9920	No
Talahua et al. [16]	MobileNetV2 + FaceNet + ANN	0.9965	Yes
Song et al. [17]	MTCNN + FaceNet + SVM/XgBoost	0.9700	Yes
Kim et al. [18]	Adaptive Margin based on Norm + ResNet	0.9751	No
Lu & Zhuang [21]	Upper-lower network + ResNet-18	0.9860	No
Ge et al. [22]	The Convolutional Visual Self-Attention Network (CVSAN)	0.9935	No
Wang et al. [23]	ConvNeXt-T + ECA	0.9976	Yes

From Table 1, in the majority of past experiments, the best model for the face mask recognition system was only found using publicly accessible face benchmark datasets. This demonstrates that real-time video deployment of the system has not yet occurred. This is a crucial stage in proving the system's effectiveness. Furthermore, the optimal hyperparameters for the dataset that yield reduced error metrics cannot be found when a person is recognized using a face mask or a classification model. Split validation may result in bias or overfitting the model when the data are typically divided into training and testing data for classification. The following research questions were proposed for this study based on the literature review:

RQ1: What technique can be used to find the best hyperparameter?

RQ2: How should the training and testing data be divided to avoid having an overfit model?

RQ3: Which classification technique is appropriate for the face verification phase?

Based on previous research and the research questions, this study intends to provide the following contributions. First, grid search is intended to perform hyperparameter tweaking in a methodical manner by automatically going through all of the sets of hyperparameter values during the model training phase. The fact that the hyperparameter settings are independent is one benefit of grid search. Second, to discover the optimal set of hyperparameters for the selected classification model, nested cross-validation is performed. This method aids in evaluating the trained models' ability to generalize accurately. This research differs from earlier studies in the face verification phase, which just employed split validation to divide the training and testing datasets, which may result in bias or overfitting the model. The nested cross-validation offers a more trustworthy standard for selecting the best model, despite being computationally time-consuming. Additionally, the best classifier methods for the face-masked verification phase are compared and determined in this study.

The remainder of this paper is structured as follows: In Section 2, the theoretical background is described. Section 3 of the proposed method is discussed. Section 4 presents the experimental setup and results. Section 5 then presents a discussion. Finally, Section 6 presents the conclusions.

2- The Theoretical Background

2-1- Multi-Task Cascaded Convolutional Networks

The facial identification method Multi-Task Cascaded Convolutional Networks (MTCNN) uses deep learning to locate and identify faces in images and videos. In the field of computer vision, it was first presented by Zhang et al. [24] and has since grown to be one of the most popular face identification techniques. A proposal network creates candidate face areas; a refined network sharpens the bounding box and facial landmarks; and an output network produces the final detection findings. Since the three stages are trained sequentially, the network can adjust its predictions at each level based on the results of the previous stage. Compared to conventional face identification algorithms, MTCNN provides a number of benefits, including the capacity to manage large-scale variations in facial appearance, the ability to handle challenging scenarios such as occlusions and non-frontal faces, as well as its speed and efficiency. MTCNN can also recognize faces at various scales, enabling it to handle faces of various sizes in a single image. Additionally, it has been utilized as a pre-processing step for various computer vision tasks, including face identification and localization of facial landmarks.

2-2- FaceNet

Schroff et al. [25] unveiled FaceNet, a system for facial identification based on deep learning. It was one of the first systems to recognize faces using a deep neural network, and it has had a significant impact on computer vision. FaceNet learns an embedding, which is a convolutional neural network-based representation of a face that may be used for comparison and recognition. The network learns to map each face image to a condensed feature vector in a high-dimensional space after being trained on a sizable dataset of face images. The similarity between facial features can then be calculated using this embedding, along with personal identification. FaceNet makes a significant contribution by training the network with a triplet loss function. Similar faces should have similar embeddings, whereas dissimilar faces should have dissimilar embeddings. This is a constraint that is enforced by the triplet loss function. This enables FaceNet to develop discriminative embedding that successfully captures the distinctive features of each face.

2-3- Hyperparameter Tuning with Grid Search

Hyperparameter tuning is the process of methodically modifying a machine learning model's hyperparameters to maximize its performance on a particular task. In contrast to model parameters, which are parameters that are learned during training, hyperparameters are parameters that are established before a model is trained. A range of values for each hyperparameter is chosen, the model is trained numerous times with various combinations of hyperparameter values, and the best set of hyperparameters is chosen based on performance evaluation on a validation set [26]. Grid search is intended to automatically go over each set of hyperparameter values while the model is being trained, allowing for systematic hyperparameter tuning [27]. In order to discover the hyperparameter configuration space, grid search is the most commonly used method. Grid search can be viewed as a method of exhaustively evaluating every conceivable combination of the hyperparameters supplied to the grid configuration [28]. Grid search operates by analyzing the Cartesian product of a user-defined, constrained set of values. Grid search alone cannot be used to further utilize the high-performing regions. As a result, the technique for manually finding the global optimum is as follows [29]:

- Start with a large search space and phase scale, then tighten them based on previous observations of successful hyperparameter choices.
- Repeat until the best outcome is attained.

2-4- Nested Cross-Validation

Nested cross-validation is a method for assessing how well machine learning models are working. It is a development of conventional cross-validation, where an inner loop is used to tune hyperparameters on a validation set and an outer loop is used to assess model performance on a test set [30]. Different combinations of hyperparameters are tested in the inner loop using a cross-validation method, and the hyperparameters that produce the best results on the validation set are chosen for the outer loop. As the validation set is held out from the training process and utilized just for hyperparameter tuning, this helps to avoid overfitting the hyperparameters to the training set. Compared to typical cross-validation, nested cross-validation ensures that the hyperparameters are selected based on their generalization performance, providing a more reliable assessment of model performance. The procedure for the nested cross-validation is given in Algorithm 1 [31] and Figure 1.

Algorithm 1 gives an implementation of the nested cross-validation where the following functions are assumed:

- createCV(data, …) creates a list of pairs (train, test) from the data.
- createGrid() creates the list of hyperparameter tuples to be tested. Grid search is employed in this paper.

- classtrain(train, theta) returns the classifier trained on train data with hyperparameters set to theta.
- accuracy(model, test) returns the accuracy (or any other quality measure) for the classifier model when run on test data.

Algorithm 1. Nested Cross-Validation

```
def nested(data, …) :
   acc_final = 0.0
   cv_outer = createCV(data, …)
   for tr_outer, te_outer in cv_outer:
      acc_max = 0.0
      for theta in createGrid(…):
         acc = 0.0
         cv_inner = createCV(tr_outer, …)
         for tr_inner, te_inner in cv_inner:
            model = classtrain(tr_inner, theta)
            acc = acc + accuracy(model, te_inner)
         if acc > acc_max:
            acc_max = acc
            theta_max = theta
      model2 = classtrain(tr_outer, theta_max)
      acc_fnal = acc_final + accuracy(model2, te_outer)
return acc_final / len(cv_outer)
```

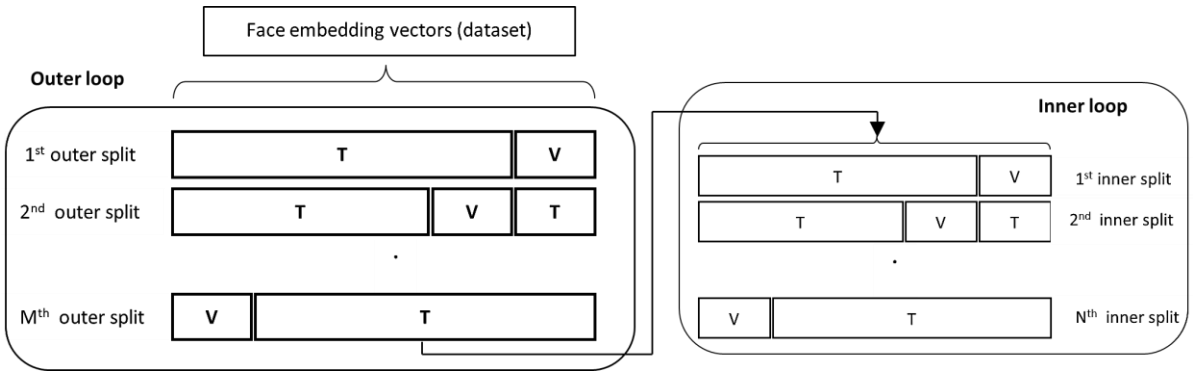


Figure 1. The nested cross-validation example. T and V represent for training and validation data

3- Proposed Methodology

Figure 2 depicts the proposed technique for recognizing a person wearing a mask. It consists of two main processes: 1) building a trained FaceNet model to extract the embeddings for the recognize between people wearing and not wearing masks. 2) Real-world deployment process.

3-1- Building a Trained FaceNet Model to Extract the Embeddings for The Recognize Between People Wearing and Not Wearing Masks

This process is divided into five steps, which are described below.

3-1-1- Face Detection using MTCNN

In this study, the MTCNN model was utilized to identify faces in an input image. Convolutional neural networks (CNNs), which are the basis of MTCNN, are used to create candidate face areas and produce a collection of bounding box proposals. It then utilizes two more CNNs to improve these bounding box proposals and categorize whether or not each proposal contains a face. To align the face within the bounding box, it uses a third CNN to conduct facial landmark detection.

3-1-2- Using Dlib 68_Face_Landmark to Locate the Mouth Part and then Wearing Mask

Dlib is a well-liked open-source library for tasks involving face landmark detection in computer vision and machine learning. It provides a pre-trained model for detecting 68 facial landmarks on a face, including the mouth. In this study, the mouth portion was extracted by Dlib as an ROI (Region of Interest), and the mouth is accessed through points. Then, the face mask image needs to be scaled to match ROI [48, 67].

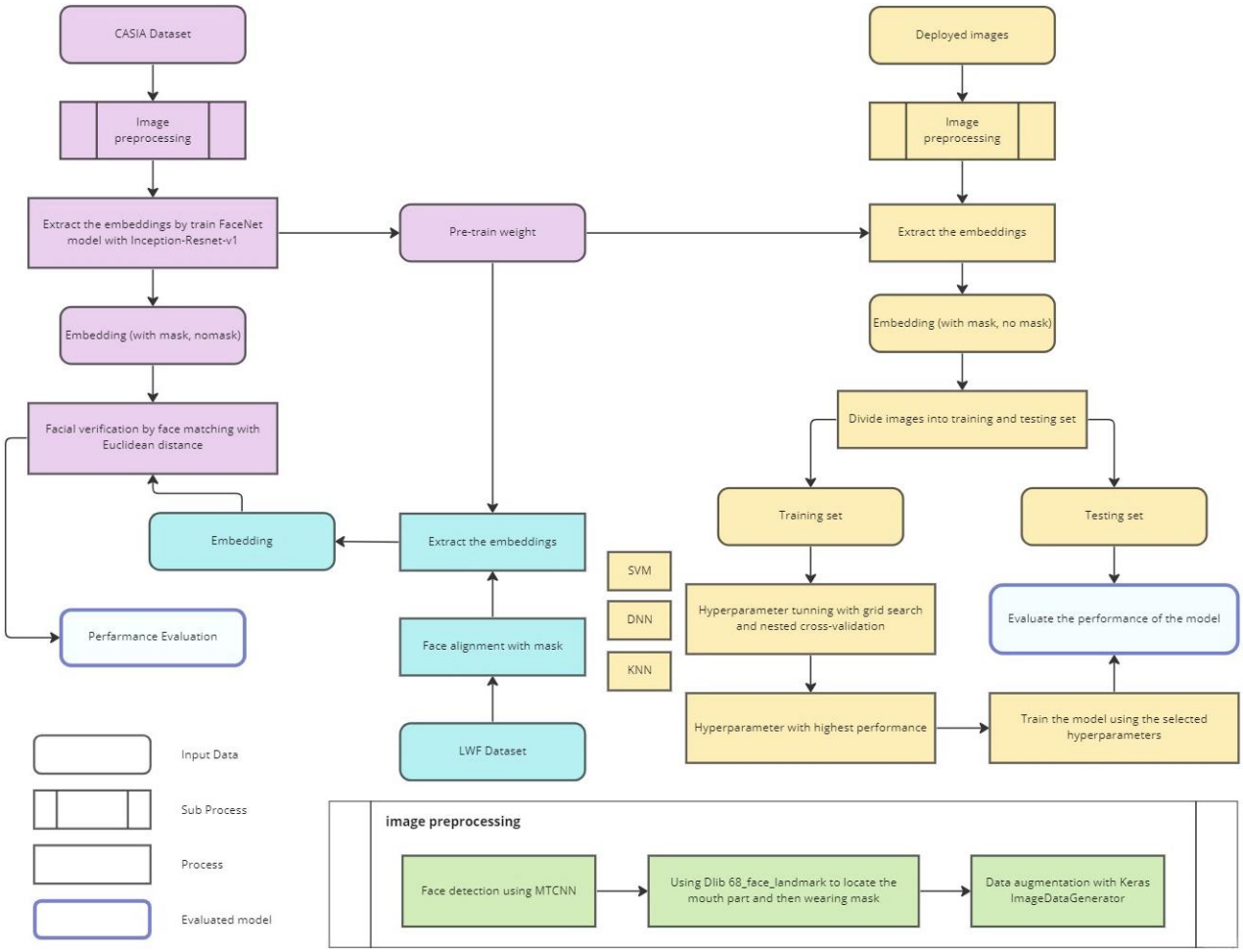


Figure 2. Proposed method for recognizing a person wearing a mask

3-1-3- Data Augmentation with Keras ImageDataGenerator

To enhance the size of the training set and decrease overfitting in deep learning models for computer vision applications, Keras ImageDataGenerator provides a wide variety of augmentation approaches [32]. This study employs horizontal flipping and brightness adjustment for both mask-wearing and without mask-wearing face images. Creating a mirror image of an image requires the horizontal flip technique, which includes turning the image horizontally along the y-axis. This method is helpful for expanding training data sets since it gives the model access to the same image from a new angle. And it can teach the model to recognize things in any orientation. Adjusting the brightness of an image is yet another method for improving the quality of captured visuals. In this study, the brightness of an image is adjusted between range [0.2,0.8]. By lowering the brightness, glare is reduced and contrast is enhanced, while increasing the brightness helps see images in low light. Under cases where images are acquired in a wide range of lighting conditions, this method can be helpful.

3-1-4- Extract the Embeddings by Train FaceNet Model with Inception-Resnet-v1

The FaceNet model is used to extract the most important features of the face with 128-dimensional embeddings from images of the face with and without a mask. Inception-Resnet-v1 [33] is a deep Convolutional Neural Network (CNN) architecture that combines Inception blocks with a residual neural network, and it is used to enhance the performance of the trained FaceNet model in this study. We then fine-tuned our trained FaceNet model by adjusting the network's parameters and pre-trained weights.

3-1-5- Facial Recognition by Face Matching with Euclidean Distance

This verification step is consolidated to recognize candidates face by performing the face matching with Euclidean distance. Face matching with Euclidean distance is a simple and effective method for comparing the similarity between two faces [10]. In order to determine the Euclidean distance between two faces, face embeddings must first be produced for each face. To determine if two faces are a match, a threshold is set for the Euclidean distance. In this study, similar faces are identified as having a distance between their embeddings that is smaller than a predetermined threshold value (0.8). When comparing two faces, if the distance is larger than a certain criterion, the faces are regarded to be distinct.

3-2- Real-World Deployment Process

We collected face image from 50 students in the classroom to create a dataset to test the efficacy of our proposed in the masked-facial recognition in real situations. After all deployed images were passed through step 3.1.1 to 3.1.4, in the verification step by face matching with Euclidean distance may not generalize well to new faces or environments that were not included in the training data, which can limit their applicability in real-world scenarios [19]. The Euclidean distance-based methods had the limited generalization ability and achieved lower accuracy rates on new faces. For this reason, this study employed in grid search for hyperparameter tuning with nested cross-validation in order to build an accurate model to verify facial features in real-world deployment process. A training set and a test set are created from the collected data, and nested cross-validation is then used to determine the optimal model hyperparameters. Model performance with varying hyperparameters is assessed in the outer loop of cross-validation, and the best hyperparameters are chosen for each fold of the outer loop in the inner loop. After averaging the results from each outer loop fold, the optimal hyperparameters are determined, and then the model is trained using these hyperparameters on the complete training set before being evaluated on the test set. The detail about this process is provided below.

- Initially, all deployed images should be divided into a 90% training set and a 10% test set. The training set will be used for hyperparameter tuning and cross-validation, while the test set will be used for model evaluation.
- Define the hyperparameter grid to search for a classification model. Using Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Deep Neural Network (DNN), this research aimed to determine which method performed best on the deployed data.
- Divide the training set to create the nested cross-outer validation's loop. Common methods for doing so include setting a fixed number of folds (k_outer=5).
- For each combination of hyperparameters.
    - Divide the current outer fold into K inner folds for nested cross-validation. We choose k_inner = 3 for the inner loop of cross-validation to perform the grid search within each outer fold.
    - Train the model on the inner folds using the current hyperparameters.
    - Evaluate the performance of the model on the outer fold using the selected hyperparameters.
- Calculate the average performance across all folds for each combination of hyperparameters. We compute the outer-loop cross-validation average accuracy score for each hyperparameter combination. Hyperparameters with the highest mean accuracy are chosen.
- Train the model using the selected hyperparameters on the entire training set.
- Evaluate the performance of the model on the test set and select the best classification model among SVM, KNN, and DNN.

4- Experimental Setup and Results

4-1- Data Description

The model was trained to extract embeddings that allow it to determine whether or not people are wearing masks. The CASIA-WebFace dataset [34] is a high-quality and diverse set of face images of humans; it contains 494,414 images with facial sub-images of roughly 10,000 people, and it was used as part of the data on which our trained FaceNet model was built. Our trained model was evaluated using the Labeled Faces in the Wild (LFW) dataset [35]. The LFW dataset contains 13,233 photographs of human faces, from which facial sub-images have been extracted in a fashion analogous to that used in the CASIA-WebFace dataset. In order to ensure that our masked facial recognition algorithm works as intended in the real world, we gathered face images from 50 students in the classroom to use as a test dataset.

4-2- Ethics Approval

Permission for the study was obtained from the ethics committee of Walailak University, Thailand (protocol no. WUEC-22-058-01).

4-3- Assessment Matrices

Performance in recognition was measured in terms of accuracy, precision, and recall in this study. Prediction results from the classification problem are summarized in the confusion matrix in Table 2. The count values summarize the proportion of correct and wrong predictions for each category. To clarify, {P, N} stands for the positive and negative testing data, while {Y, N} stands for the classifier predictions for the positive and negative classes [36].

Table 2. Confusion matrix

	Actual Positive (P)	**Actual Negative (N)**
Predicted Positive (Y)	TP (true positives)	FP (false positives)
Predicted Negative (N)	FN (false negatives)	TN (true negatives)

Number of correct predictions for a positive example is denoted by "true positive" (TP), for negative examples by "true negative" (TN), for incorrect predictions for a positive example by "false positive" (FP), and for incorrect predictions for a negative example by "false negative" (FN). Equations 1 to 4 below define the assessment measures used to evaluate the performance of recognition from the datasets.

Accuracy:

$$\frac{TP+TN}{TP+FP+FN+TN} \tag{1}$$

Precision:

$$\frac{TP}{TP+FP}, \tag{2}$$

Recall:

$$\frac{TP}{TP+FN}, \tag{3}$$

F-measure:

$$\frac{(1+\beta^2)\times Recall\times Precision}{\beta^2\times Recall+ Precision}, \tag{4}$$

where β is a coefficient used to adjust the relative importance of precision versus the recall, which is usually set to 1. The F-measure is high when both recall and precision are high, indicating the goodness of a learning algorithm for the interest class [37].

4-4- Experimental Configuration

The experimental platform operating system was Windows 10, the GPU was an NVIDIA GeForce RTX 3060 Ti 8GB, CUDA Core 4,864, GPU Clock 1665 MHz with 32GB Memory. Python is the language used for development. To train models, we use the deep learning tools Keras, Tensorflow, and scikit learn.

4-5- Experimental Results

We employed the CASIA-WebFace dataset as part of the training data for our deep learning model, which is typically trained on large-scale datasets. Furthermore, the Labeled Faces in the Wild (LFW) dataset was used to test our trained model. The process of data preprocessing is;

- *Face detection and mask coverage:* In order to extract only face images for CASIA-WebFace, we pre-processed them with MTCNN. It is challenging for a model to learn the feature mapping when a face is occluded by a mask due to the scarcity of masked-face datasets, which leads to a low recognition rate. To solve this problem, we used the CASIA-WebFace dataset to generate masked-face images. We integrate MTCNN with Dlib, which enables us to quickly and precisely extract facial features. After applying MTCNN, the Dlib library is utilized to identify 68 facial landmarks on a person's face, which include the mouth. The specific area of the mouth can be accessed by using the coordinates [48, 67]. After that, the mask image must be resized to fit the ROI. One of the mask templates is randomly placed over the mouth region to produce a masked-face image (a surgical mask, a KN95 mask, and a black mask). The result is shown in Figure 3.

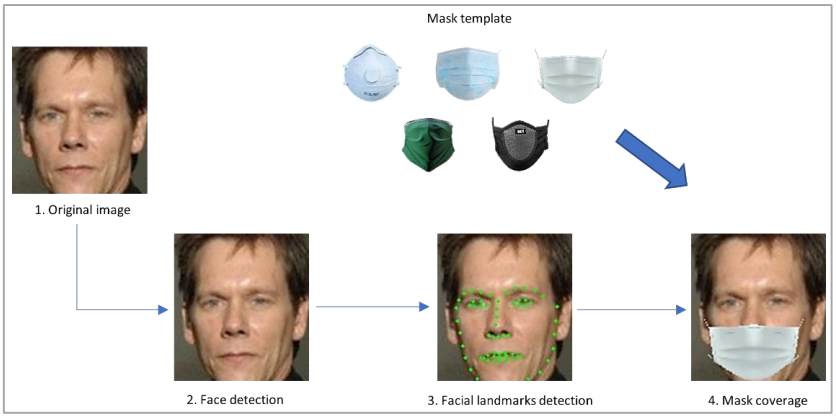


Figure 3. Face detection and mask coverage

- *Data augmentation:* Our goals for data augmentation are to increase the size and diversity of the raw datasets and decrease overfitting problems, as the quality and variety of the training dataset are vital to improving the performance and ability of the model to generalize. Because of this, we used Keras ImageDataGenerator's horizontal flip and brightness adjustment features, which yielded an adjusted brightness of between 0.2 and 0.8. Figure 4 depicts some examples of ImageDataGenerator-processed augmented images. There are a total of 612948 images, 306474 of which are unmasked and 306474 of which are masked.

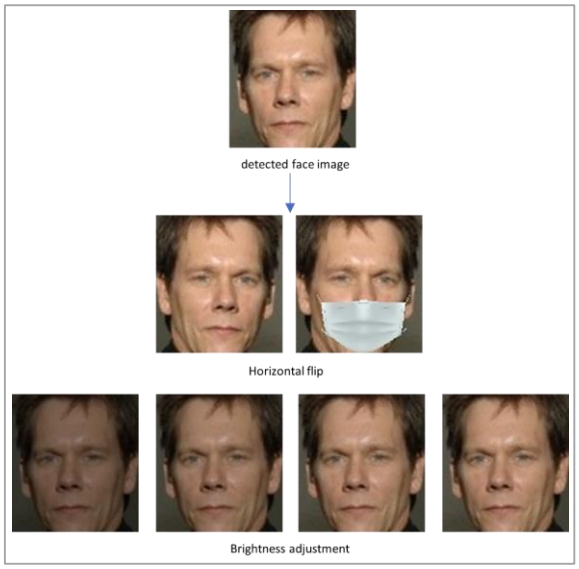


Figure 4. Example of augmented images

The FaceNet model is used to extract the most essential elements of face images, and it returns a vector of 128 features. The image is fed into the network, processed by the neural network, and output with an embedding of each face [16] indicated by $f(x)\ \epsilon\ R^d$. The goal of this technique is to move one person's image (x_i^a) closer to all other images of that person (x_i^p) and farther from any other images of people (x_i^n). The calculation of the loss (L) is demonstrated by Equation 5, which includes the use of a margin (α) between positive and negative pairs.

$$L = \sum_i^N \left[\left\|(x_i^a) - f(x_i^p)\right\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\right], \quad (5)$$

The input is a 112×112 pixel image, and the output is a 128-elements "face embedding" vector containing information on the face in the image. The Inception-ResNet V.1 architecture is the backbone of our trained FaceNet model. Finding an optimal set of training settings and initial weights is the subsequent stage. As a loss function, we employ the Adam optimizer with a learning rate of 0.0005 and the categorical cross-entropy. The batch size is 128, and there are 50 epochs in all, with an average epoch period of about 48.5 minutes.

As test data, the LFW datasets are employed. From the LFW dataset, we randomly selected 1000 images from various classes. Use face masks in certain images. Face embeddings were calculated using our trained FaceNet model. Face encodings from a subset of the LFW dataset are compared to the encodings of all images in the training dataset using the Euclidean distance. According to the results of this study, comparable faces are those whose embedding distance is less than a specified threshold value (0.8). It has been determined that two faces are considered to be different if the distance between them exceeds a predetermined threshold. On LFW datasets, our trained FaceNet model achieves an accuracy of 0.99817. Table 3 provides a concise summary of the FaceNet model efficiency and parameter tuning.

Table 3. FaceNet model efficiency and parameter tuning

Parameter	Detail	Accuracy
Learning rate	0.0005	
Epochs	50	
Batch size	128	0.99817 (randomly selected 1000 images with mask from LWF dataset)
Optimizer	Adam	
Loss function	Categorical cross-entropy	

4-6- Real-World Experiment

To ensure our method works as intended for masked facial recognition in the real world, we gathered face images from 50 students in the classroom to use as a dataset. For creating the training and testing dataset for the verification phase model, MTCNN is used for face detection on 50 student images without masks. The Dlib library is used to extract 68 facial landmarks, including the mouth, from a face in order to create the masked-face image dataset. Finally, a masked-face image is created by randomly placing one of the mask templates over the mouth area and resizing the mask image to fit the ROI. Accordingly, after running the face detection procedure, we will get 50 images of students without masks and another 50 of the same students wearing masks. The result is shown in Figure 5. As part of the data augmentation step, we applied horizontal flip and brightness adjustments to all images. As a result, following data augmentation, we have 500 total images, 250 without masks and 250 with masks.

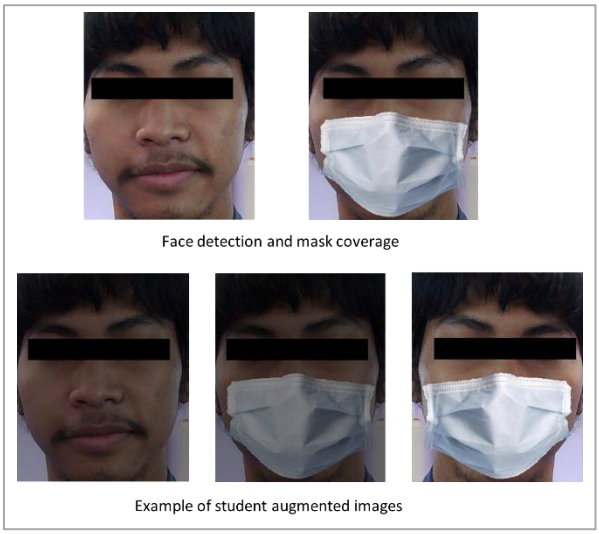


Figure 5. Example of face detection, mask coverage, and some images from student image augmentation

One of the major drawbacks of using Euclidean distance for face matching in the verification step is that it is sensitive to the feature set used to represent the face; if the features are not comprehensive or discriminative enough, the Euclidean distance may fail to accurately distinguish between faces, resulting in poor recognition performance that can negatively affect the accuracy and reliability of real-world face recognition applications. After that, all images were embedded using our pre-trained FaceNet model, which provides a 128-element feature vector. Therefore, in the verification step, once the feature vector has been obtained, any machine learning classification model can be applied. In this case, we compared the performance of SVM, KNN, and DNN models using a grid search with hyperparameter tuning and nested cross-validation. This approach differs from other research studies that only use a single machine learning model for classification and typically do not report the best parameter values. The dataset contains 500 images, which were split into a 90% training set and a 10% test set. The training set was utilized for conducting hyperparameter tuning and cross-validation, while the test set was reserved for evaluating the final model. To perform nested cross-validation with grid search, the training set is divided into outer and inner folds. The outer fold is used for evaluating the model's performance, while the inner folds are used for hyperparameter tuning through grid search. The model is trained on the inner folds using selected hyperparameters and evaluated on the outer fold to obtain the mean accuracy for each set of hyperparameters. The hyperparameters with the highest mean accuracy are then chosen to train the model on the entire training set. Finally, the performance of SVM, KNN, and DNN models is compared on the test set to select the best-performing model. According to Table 4, the optimal parameters for the SVM, KNN, and DNN models have been identified, and their corresponding accuracy, precision, and recall metrics have been reported in Table 5. Based on the results, it can be concluded that the SVM model outperformed the other models in terms of recognition accuracy of 0.99912. The SVM model was optimized using a polynomial kernel, a C value of 1000.0, a gamma value of 0.1, and a maximum iteration of 40.

Table 4. The optimal parameter for SVM, KNN, and DNN

Classifier	Optimal parameter	Detail
SVM	Kernel function	Polynomial
	Regularization parameter (“C”)	1000.0
	Gamma	0.1
	Max_iter	40
KNN	Leaf_size	5
	n_neighbors	3
	Weights	Uniform
	Metric	Minkowski
DNN	No. of neurons in the initial layer	128
	No. of neurons in a hidden layer with built-in ReLu	115
	No. of neurons in a hidden layer with built-in ReLu	110
	No. of neurons in a hidden layer with built-in ReLu	80
	No. of neurons in the output layer with a Softmax	50
	Epochs	150
	Batch sizes	20
	Optimizer	rmsprop
	Loss function	Categorical cross-entropy

Table 5. The optimal parameter for SVM, KNN, and DNN

Classifier	Class	Accuracy	Precision	Recall	F-measure
SVM	Without mask	0.99912	0.99925	0.99893	0.99941
	With mask		0.98417	0.97934	0.98411
KNN	Without mask	0.98491	0.98141	0.98381	0.98479
	With mask		0.95990	0.93723	0.95201
DNN	Without mask	0.99544	0.99305	0.98883	0.99331
	With mask		0.98041	0.97415	0.98164

5- Discussion

Proposed solutions from the research questions and discussion of facial and masked facial recognition are outlined below:

RQ1: What technique can be used to find best hyperparameter?

In this study, the use of a grid search has proven useful. Grid search is a computationally expensive approach for hyperparameter tuning as it produces a model for each set of provided hyperparameters and then assesses each model. Despite this, it is still widely used because it determines the optimal approach to adjusting the hyperparameters based on the training set. It is acceptable for the training phase to take longer because it is conducted offline before testing, which allows the network to test every potential combination and find the best-performing one. This finding is consistent with the studies conducted by Saad et al. [28] and Khamparia & Singh [38]. We opted to use grid search to ensure the best possible accuracy during the testing phase rather than reduce complexity and training time. This precision is essential for recognizing individuals wearing masks. We chose grid search because of its simplicity, higher accuracy, and greater reliability, as well as its ability to work in high-dimensional spaces.

RQ2: How should the training and testing data be divided to avoid having an overfit model?

We applied nested cross-validation in our work. It is a beneficial technique for model selection and hyperparameter tuning because it helps to avoid overfitting and provides an unbiased estimate of model performance [39]. In nested cross-validation, the data is divided into multiple folds, with each fold being used as a separate validation set for the other folds. This process is repeated multiple times, with different folds being used as validation sets each time, and the average performance across all iterations is used as the final estimate of model performance. For our work, we decided to combine grid search with hyperparameter tuning and nested cross-validation. The reason for this is that the utilization of these techniques can enhance the model's stability against data variations and prevent overfitting. Despite the fact that grid search with hyperparameter tuning can be time-consuming, the implementation of nested cross-validation can effectively reduce the computational burden by optimizing the data usage during hyperparameter tuning. The combination of grid search with hyperparameter tuning and nested cross-validation yielded an accuracy of 0.99912 for recognition, whereas using 10-fold cross-validation without grid search resulted in an accuracy of 0.98394. This clearly indicates that the use of grid search with hyperparameter tuning and nested cross-validation can significantly enhance the model's performance, generalization, and robustness while effectively optimizing the utilization of data.

RQ3: Which classification technique is appropriate for face verification phase?

In this study, three classifiers, namely SVM, KNN, and DNN, were evaluated for their performance in face mask recognition. The findings demonstrated that SVM exhibited the most optimal results. SVM is an efficient algorithm for classification tasks, specifically face recognition, as it maps data points to a high-dimensional space and identifies the most suitable hyperplane that divides them into different classes. Due to its capability to classify images using extracted features, SVM is an appropriate choice for face mask recognition. Moreover, SVM's performance can be significantly influenced by its various hyperparameters, such as kernel type, regularization parameter, and gamma value. Thus, grid search can be employed to adjust these hyperparameters to improve the performance of the SVM algorithm in face mask recognition. Grid search aims to experiment with diverse combinations of hyperparameters and select the most effective one, resulting in enhanced accuracy and robustness of the model.

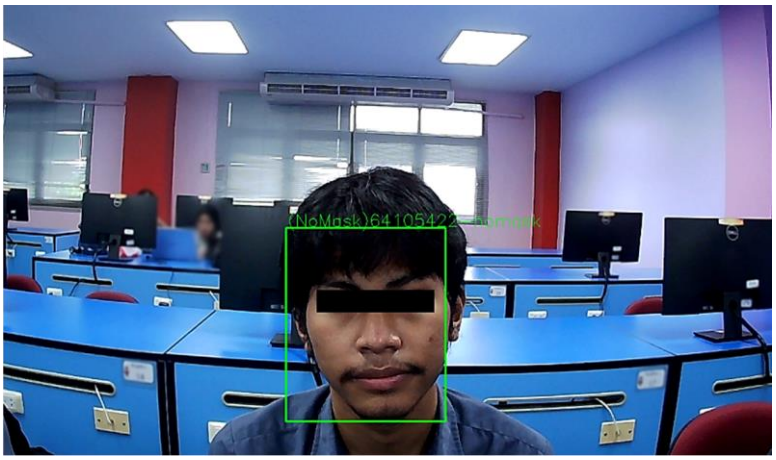
We have presented a comparison of existing face recognition systems in terms of their accuracy in recognizing faces when they are wearing masks. The comparison Table 6 includes information on the datasets used, the number of images in each dataset, and the face recognition techniques used. Each system used a different approach to accurately recognize faces from images. This comparison helps to assess the performance of various face recognition systems in dealing with the challenge of face masks.

Table 6. A comparison of current methods based on the accuracy of face-masked recognition

Method	Technique	Accuracy	Dataset	No. of image in dataset	Real-world deployment	Image or video capture distance (Meters)
Ejaz et al. [5]	Viola-Jones algorithm + PCA	0.8350	ORL database +user dataset	500	No	No
Maharani et al. [6]	VGG16 + FaceNet	1.0000	User dataset	600	Yes	1.0-1.5
Montero et al. [7]	MTArcFace	0.9978	MS1MV2	Not specified	No	No
Hariri [8]	VGG16 + BoF + MLP	0.9130	RMFRD and SMFRD	5000 and 50000	No	No
Golwalkar et al. [10]	FaceMaskNet-21	0.8892	User dataset	2000	Yes	Not specified
Ejaz et al. [12]	MTCNN + FaceNet + SVM	0.9850	AR Face Database + IIIT-Delhi Disguise Version 1 Face Database +User dataset	Not specified	No	No
Moungsouy et al. [15]	MTCNN + Inception-ResNet-v1	0.9920	CASIA and LWF	494,414 and 13,233	No	No
Talahua et al. [16]	MobileNetV2 + FaceNet + ANN	0.9965	User dataset	13,359	Yes	Not specified
Song et al. [17]	MTCNN + FaceNet + SVM/XgBoost	0.9700	CASIA and Masked Face Net	14,646	Yes	Not specified
Kim et al. [18]	Adaptive Margin based on Norm + ResNet	0.9751	MS1MV2, MS1MV3, WebFace4M and LFW	More than 1 million images	No	No
Lu & Zhuang [21]	Upper-lower network + ResNet-18	0.9860	Randomly select from search engines	200	No	No
Ge et al. [22]	The Convolutional Visual Self-Attention Network (CVSAN)	0.9935	VGGFace2 and LWF	More than 3 million images	No	No
Wang et al. [23]	ConvNeXt-T + ECA	0.9976	CASIA + LWF + User dataset	494,414 and 13,233	Yes	Not specified
Proposed method	**FaceNet + optimized SVM**	**0.99912**	**CASIA + LWF + User dataset**	**612,948 and 1,000**	**Yes**	**1.0**

We used grid search with nested cross-validation to find the best hyperparameters in the verification step, which resulted in a recognition accuracy of 0.99912. Furthermore, we used a camera to collect data for recognition. The recognition results are shown in Figure 6, and it can be seen that if the person with the camera is in the database, the student ID and a label of "Mask" or "No Mask" are placed on the green bounding box. Table 6 shows that only five techniques have been employed in real-world scenarios. Although Maharani et al. [6] achieved an accuracy of 1.0000, they did not specify the optimal hyperparameters used. Additionally, their study employed a small number of user datasets. Golwalkar et al. [10] presented the FaceMaskNet-21 model using a user dataset of 2000 images, which achieved an accuracy of only 0.8892. This performance is lower than our proposed method. Talahua et al. [16] introduced a model used MobileNetV2 + FaceNet + ANN that was trained on a dataset of 13,359 images and achieved an impressive accuracy of 0.9965. This outperformed our own method. However, their approach splits the data into 80% for training and 20% for testing, which can lead to bias or overfitting. Our method, on the other hand, uses nested cross-validation to divide the data for training and testing, avoiding these potential issues.

Song et al. [17] did not specify how their training and testing data was divided, and they did not mention the hyperparameters used in the SVM and XgBoost classifiers. This approach differs from our proposed method, which identifies optimized hyperparameters using grid search. Wang et al. [23] introduced the ConvNeXt-T + ECA model, which achieved an accuracy of 0.9976. However, they did not disclose their approach to dividing the training and testing data, nor did they specify the hyperparameters employed in the verification phase. This could result in model bias or overfitting, and inaccurate verification results when new images of individuals are introduced. This is unlike our proposed method, which employs nested cross-validation with hyperparameter tuning. This helps to prevent the hyperparameters from being overfitted to the training sets, and guarantees that they are chosen based on their generalization performance, resulting in a more dependable model evaluation.



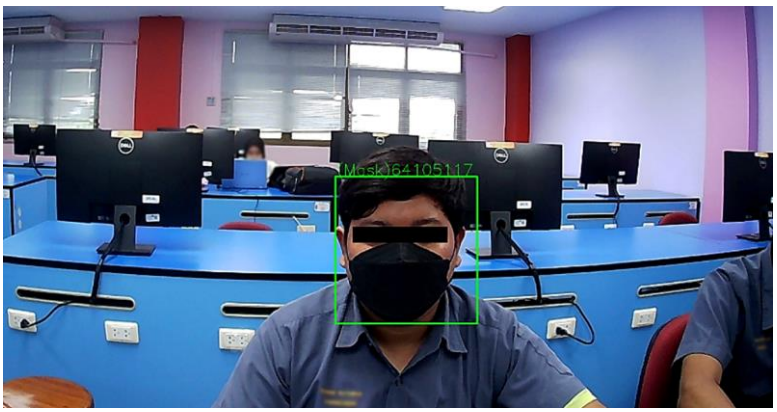


Figure 6. Example of real-world result

6- Conclusion

Although COVID-19 cases have slightly decreased, it is still crucial to wear face masks, emphasizing the need for developing face recognition systems capable of identifying masked faces. However, previous methods have only utilized publicly available benchmark datasets to create face mask recognition models, which have not been adequately evaluated in real-time video settings. Additionally, identifying the optimal hyperparameters for face mask recognition models is challenging since standard classification models struggle to recognize people wearing masks. To tackle these issues, our study proposes an approach that incorporates grid search with hyperparameter tuning and nested cross-validation during the verification phase. Unlike previous studies that only used single machine learning models for classification and did not report optimal parameter values, our approach is innovative. Our results show that the SVM model with hyperparameter tuning outperforms other models, achieving a recognition accuracy of 0.99912. Importantly, our work is relevant in real-life masked-face recognition scenarios, verifying the system's capacity to identify the masked person's identity through facial recognition.

The main limitation of our proposed method is the computation time. Although using a grid search with hyperparameter tuning and nested cross-validation can significantly improve the recognition performance, it still takes longer to process compared to using the traditional classifier without grid search or nested cross-validation. Therefore, this is a motivation for future work to apply the method proposed by Fayed et al. [40] to speed up the grid search for optimal parameter selection for SVM. Other optimization hyperparameter methods, such as Bayesian optimization hyperparameters, can also be used. In addition, we will develop our proposed method into an application for the attendance management system, which will be managed by the Center for Educational Services at Walailak University.

7- Declarations

7-1- Author Contributions

Conceptualization, P.T. and S.S.; methodology, P.T.; software, P.T. and P.K.; validation, P.T. and S.S.; formal analysis, P.T.; investigation, P.T.; resources, S.S. and P.K.; data curation, P.T. and S.S.; writing—original draft preparation, P.T., S.S., and D.N.M.N.; writing—review and editing, P.T., S.S., and D.N.M.N.; visualization, P.T.; supervision, S.S.; project administration, S.S. All authors have read and agreed to the published version of the manuscript.

7-2- Data Availability Statement

Data sharing is not applicable to this article.

7-3- Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

7-4- Acknowledgements

The authors express their gratitude for the support provided by The Center for Educational Services, Walailak University.

7-5- Institutional Review Board Statement

Permission for the study was obtained from the ethics committee of Walailak University, Thailand (Protocol No. WUEC-22-058-01).

7-6- Informed Consent Statement

Informed consent was obtained from all students involved in the study.

7-7- Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this manuscript. In addition, the ethical issues, including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancies have been completely observed by the authors.

8- References

[1] Khel, M. H. K., Kadir, K., Albattah, W., Khan, S., Noor, M. N. M. M., Nasir, H., ... & Khan, A. (2021). Real-time monitoring of COVID-19 SOP in public gathering using deep learning technique. Emerging Science Journal, 5(Special issue), 182-196. doi:10.28991/esj-2021-SPER-14.

[2] Damer, N., Boutros, F., Süßmilch, M., Kirchbuchner, F., & Kuijper, A. (2021). Extended evaluation of the effect of real and simulated masks on face recognition performance. IET Biometrics, 10(5), 548–561. doi:10.1049/bme2.12044.

[3] Alzu'bi, A., Albalas, F., Al-Hadhrami, T., Younis, L. B., & Bashayreh, A. (2021). Masked face recognition using deep learning: A review. Electronics (Switzerland), 10(21), 2666. doi:10.3390/electronics10212666.

[4] Ullah, N., Javed, A., Ali Ghazanfar, M., Alsufyani, A., & Bourouis, S. (2022). A novel DeepMaskNet model for face mask detection and masked facial recognition. Journal of King Saud University - Computer and Information Sciences, 34(10), 9905–9914. doi:10.1016/j.jksuci.2021.12.017.

[5] Ejaz, Md. S., Islam, Md. R., Sifatullah, M., & Sarker, A. (2019). Implementation of Principal Component Analysis on Masked and Non-masked Face Recognition. 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT). doi:10.1109/icasert.2019.8934543.

[6] Maharani, D. A., Machbub, C., Rusmin, P. H., & Yulianti, L. (2020). Improving the Capability of Real-Time Face Masked Recognition using Cosine Distance. 2020 6th International Conference on Interactive Digital Media (ICIDM), Bandung, Indonesia. doi:10.1109/icidm51048.2020.9339677.

[7] Montero, D., Nieto, M., Leskovsky, P., & Aginako, N. (2022). Boosting Masked Face Recognition with Multi-Task ArcFace. 2022 16th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), Dijon, France. doi:10.1109/sitis57111.2022.00042.

[8] Hariri, W. (2022). Efficient masked face recognition method during the COVID-19 pandemic. Signal, Image and Video Processing, 16(3), 605–612. doi:10.1007/s11760-021-02050-w.

[9] Boutros, F., Damer, N., Kirchbuchner, F., & Kuijper, A. (2022). Self-restrained triplet loss for accurate masked face recognition. Pattern Recognition, 124, 108473. doi:10.1016/j.patcog.2021.108473.

[10] Golwalkar, R., & Mehendale, N. (2022). Masked-face recognition using deep metric learning and FaceMaskNet-21. Applied Intelligence, 52(11), 13268–13279. doi:10.1007/s10489-021-03150-3.

[11] Hong, J. H., Kim, H., Kim, M., Nam, G. P., Cho, J., Ko, H.-S., & Kim, I.-J. (2021). A 3d Model-Based Approach For Fitting Masks To Faces In The Wild. 2021 IEEE International Conference on Image Processing (ICIP), Anchorage, AK, USA. doi:10.1109/icip42928.2021.9506069.

[12] Ejaz, Md. S., & Islam, Md. R. (2019). Masked Face Recognition Using Convolutional Neural Network. 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI). doi:10.1109/sti47673.2019.9068044.

[13] Deng, H., Feng, Z., Qian, G., Lv, X., Li, H., & Li, G. (2021). MFCosface: a masked-face recognition algorithm based on large margin cosine loss. Applied Sciences (Switzerland), 11(16), 7310. doi:10.3390/app11167310.

[14] Li, Y., Guo, K., Lu, Y., & Liu, L. (2021). Cropping and attention based approach for masked face recognition. Applied Intelligence, 51(5), 3012–3025. doi:10.1007/s10489-020-02100-9.

[15] Moungsouy, W., Tawanbunjerd, T., Liamsomboon, N., & Kusakunniran, W. (2022). Face recognition under mask-wearing based on residual inception networks. Applied Computing and Informatics, 1-14. doi:10.1108/ACI-09-2021-0256.

[16] Talahua, J. S., Buele, J., Calvopina, P., & Varela-Aldas, J. (2021). Facial recognition system for people with and without face mask in times of the covid-19 pandemic. Sustainability (Switzerland), 13(12), 6900. doi:10.3390/su13126900.

[17] Song, Z., Nguyen, K., Nguyen, T., Cho, C., & Gao, J. (2022). Spartan Face Mask Detection and Facial Recognition System. Healthcare (Switzerland), 10(1), 87. doi:10.3390/healthcare10010087.

[18] Kim, M., Jain, A. K., & Liu, X. (2022). AdaFace: Quality Adaptive Margin for Face Recognition. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr52688.2022.01819.

[19] Deng, J., Guo, J., Xue, N., & Zafeiriou, S. (2019). ArcFace: Additive Angular Margin Loss for Deep Face Recognition. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2019.00482.

[20] Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., & Liu, W. (2018). CosFace: Large Margin Cosine Loss for Deep Face Recognition. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. doi:10.1109/cvpr.2018.00552.

[21] Lu, H., & Zhuang, Z. (2022). ULN: An efficient face recognition method for person wearing a mask. Multimedia Tools and Applications, 81(29), 42393–42411. doi:10.1007/s11042-022-13495-7.

[22] Ge, Y., Liu, H., Du, J., Li, Z., & Wei, Y. (2023). Masked face recognition with convolutional visual self-attention network. Neurocomputing, 518, 496–506. doi:10.1016/j.neucom.2022.10.025.

[23] Wang, Y., Li, Y., & Zou, H. (2023). Masked Face Recognition System Based on Attention Mechanism. Information (Switzerland), 14(2), 87. doi:10.3390/info14020087.

[24] Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. IEEE Signal Processing Letters, 23(10), 1499–1503. doi:10.1109/LSP.2016.2603342.

[25] Schroff, F., Kalenichenko, D., & Philbin, J. (2015). FaceNet: A unified embedding for face recognition and clustering. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). doi:10.1109/cvpr.2015.7298682.

[26] Elgeldawi, E., Sayed, A., Galal, A. R., & Zaki, A. M. (2021). Hyperparameter tuning for machine learning algorithms used for arabic sentiment analysis. Informatics, 8(4), 79. doi:10.3390/informatics8040079.

[27] Jiang, X., & Xu, C. (2022). Deep Learning and Machine Learning with Grid Search to Predict Later Occurrence of Breast Cancer Metastasis Using Clinical Data. Journal of Clinical Medicine, 11(19), 5772. doi:10.3390/jcm11195772.

[28] Saad, M. H., Hashima, S., Sayed, W., El-Shazly, E. H., Madian, A. H., & Fouda, M. M. (2023). Early Diagnosis of COVID-19 Images Using Optimal CNN Hyperparameters. Diagnostics, 13(1), 76. doi:10.3390/diagnostics13010076.

[29] Zöller, M. A., & Huber, M. F. (2021). Benchmark and Survey of Automated Machine Learning Frameworks. Journal of Artificial Intelligence Research, 70, 409–472. doi:10.1613/JAIR.1.11854.

[30] Tahkola, M., & Guangrong, Z. (2022). ATSC-NEX: Automated Time Series Classification With Sequential Model-Based Optimization and Nested Cross-Validation. IEEE Access, 10, 39299–39312. doi:10.1109/ACCESS.2022.3166525.

[31] Wainer, J., & Cawley, G. (2021). Nested cross-validation when selecting classifiers is overzealous for most practical applications. Expert Systems with Applications, 182, 115222. doi:10.1016/j.eswa.2021.115222.

[32] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1). doi:10.1186/s40537-019-0197-0.

[33] Peng, S., Huang, H., Chen, W., Zhang, L., & Fang, W. (2020). More trainable inception-ResNet for face recognition. Neurocomputing, 411, 9–19. doi:10.1016/j.neucom.2020.05.022.

[34] Yi, D., Lei, Z., Liao, S., & Li, S. Z. (2014). Learning Face Representation from Scratch. doi:10.48550/arXiv.1411.7923.

[35] Huang, G. B., Mattar, M., Berg, T., & Learned-Miller, E. (2008). Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Workshop on faces in 'Real-Life' Images: detection, alignment, and recognition, 17-20 October, 2008, Marseille, France.

[36] Kaur, A., & Kaur, I. (2018). An empirical evaluation of classification algorithms for fault prediction in open source projects. Journal of King Saud University - Computer and Information Sciences, 30(1), 2–17. doi:10.1016/j.jksuci.2016.04.002.

[37] Chen, S., He, H., & Garcia, E. A. (2010). RAMOBoost: Ranked minority oversampling in boosting. IEEE Transactions on Neural Networks, 21(10), 1624–1642. doi:10.1109/TNN.2010.2066988.

[38] Khamparia, A., & Singh, K. M. (2019). A systematic review on deep learning architectures and applications. Expert Systems, 36(3), e12400. doi:10.1111/exsy.12400.

[39] Krstajic, D., Buturovic, L. J., Leahy, D. E., & Thomas, S. (2014). Cross-validation pitfalls when selecting and assessing regression and classification models. Journal of Cheminformatics, 6(1), 1–15. doi:10.1186/1758-2946-6-10.

[40] Fayed, H. A., & Atiya, A. F. (2019). Speed up grid-search for parameter selection of support vector machines. Applied Soft Computing Journal, 80, 202–210. doi:10.1016/j.asoc.2019.03.037.