Journal of Information Systems and Informatics
Vol. 6, No. 2, June 2024 e-ISSN: 2656-4882 p-ISSN: 2656-5935
DOI: 10.51519/journalisi.v6i2.759 Published By DRPM-UBD

Realtime-Based System for Facemask Detection Using PCA, with CNN and COCO Model

Ubong Etuk[1,*], Imeh Umoren[2], Odudu Umoren[3], Saviour Inyang[4]

[1]School of Computing, University of Glasgow, Glasgow G12 8QQ, UK
[2,3,4] Department of Computer Science, Akwa Ibom State University, Mkpat Enin, Nigeria.
Email: u.etuk.1@research.gla.ac.uk[1*], imehumoren@aksu.edu.ng[2], eumoreno1111@gmail.com[3], sinyang5672@gmail.com[4]

Abstract

The instant spread of COVID-19 has underscored the need for effective measures such as wearing face masks to control transmission. As a response, facemask detection systems using advanced machine learning techniques have become essential for ensuring compliance and public safety. This research focused on developing a system for detecting facemask usage using a hybridized approach comprising of Convolutional Neural Networks (CNN), Principal Component Analysis (PCA), and the Common Objects in Context (COCO) model. A hybridized detection model is often explored to enhance the precision and efficiency of previous methods that leveraged traditional machine learning or deep learning for the same task. Hence, this system effectively identifies whether individuals are properly wearing masks, not wearing masks at all, or wearing masks improperly from images and real-time video streams using bounding boxes. The results demonstrate that the hybrid approach achieves high accuracy in detecting various facemask conditions across different scenarios. Evaluation metrics such as Average Precision (AP) and Average Recall (AR) indicate the model's robustness, with a reported AP value of 70% and an AR value of 81%, primarily evaluated on larger objects within images. Further evaluations involving different individuals and types of facemasks revealed variability in detection accuracy, highlighting the model's effectiveness and areas for improvement. Nevertheless, the development and deployment of facemask detection systems are crucial for managing public health and ensuring safety in the face of ongoing and future pandemics.

Keywords: Machine Learning (ML), Deep neural learning (DL), Convolutional Neural Network (CNN), Principal Component Analysis (PCA), Facemask

1. INTRODUCTION

The instant spread of COVID-19 virus has resulted in an urgent need for effective measures to control its transmission, including wearing face masks to prevent the virus from spreading through respiratory droplets. The coronavirus was first discovered in Wuhan, China [2]. By the end of April 2023, over 455 million confirmed cases and 6.2 million deaths had been recorded globally. The World

This work is licensed under a Creative Commons Attribution 4.0 International License.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

Health Organization declared it a global pandemic on March 11, 2020 [1] [2]. COVID-19 has affected over 222 countries and has resulted in millions of deaths. The covid-19 virus spreads through air channels such as saliva droplets or nasal discharge emitted when an infected individual coughs or sneezes, making it highly contagious. Hence, to control its transmission, it has become important to use face masks, which are designed to cover the wearer's nose and mouth, leaving the eyes and other facial features visible [3].

Facemasks come in various types, manufactured from a range of materials, each serving different purposes and filtration capabilities [2]. These include surgical masks, KN95 masks, N95 masks, FFP (1-3) masks, respirator masks, sponge masks, reusable cloth masks, and active carbon masks. Besides, these facemasks can be worn incorrectly, thereby undermining their protective capacity as a barrier between the wearer's respiratory system and the external environment (see Figure 1). In response to the pandemic, many public and private facilities made it mandatory for individuals to wear the required facemasks, using security personnel to ensure compliance. To effectively monitor facemask compliance, systems were developed that leverage on machine-learning approach [4] for facemask detection. Machine Learning lets computers analyze data and study from it without being explicit programmed. The application of machine learning in facemask detection tasks typically involves analyzing the digital image to identify facial features, extracting them, and then using specific algorithms to train the processed input data to determine the learning outcome.

Furthermore, detecting facemask in a digital image often involves two processes: detecting a face that is present in the image and classifying whether detected face is wearing face mask. Different approaches have been explored for such detection tasks, including traditional machine learning-based approaches [5] [6], deep learning-based approaches [7] [8] [9] [10] [11], and hybrid approaches [12] [13] [14]. The traditional ML approaches often adopt handcrafted feature extraction, while the DL-based models automatically learn and extract the features based on the multiple neural network layers, hence making them more effective than traditional ones. Moreover, existing works [7] [8] [9] [10] [11] focus more on the use of DL-based algorithms such as Convolutional Neural Networks (CNN), for identifying users with facemasks but these algorithms also have their demerits which includes, lacking robustness for low-quality images and failing to capture long-range dependencies. Studies that explored the hybridized approach, which combines both DL and traditional models, were thus designed to enhance the precision and efficiency of current detection models [12] [13].

This research focuses on identifying users with facemasks, without facemasks, and with incorrect facemasks from an image with the aid of hybrid Machine Learning method. This hybridized approach involves using CNN, PCA and COCO for the facemask detection tasks. CNN will be used for feature extraction, leveraging on

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

its deep learning capabilities to automatically distinguish features from input digital images. PCA will be employed for feature selection to minimize the dimensionality of the extracted data feature space (i.e. large feature vectors) while retaining essential information for accurate detection. COCO will be used in model training to enhance the recognition of faces wearing masks, faces not wearing facemasks, and with incorrectly worn facemasks. Moreover, COCO is a large collection of objects widely used in the domain of computer vision for object detection, segmentation, captioning [16]. Object detection is a technique that identifies the presence, location, and type of one or more objects in digital images or videos where the position of the object is usually indicated by a bounding box.

Object detection algorithms or systems often make use of machine learning or deep learning to effectively detect instances of objects. Paul Viola and Michael Jones introduced a popular object detection framework for face detection [15]. The Viola-Jones algorithm revolutionized real-time object detection and is used in many object detection systems such as in [19], by proposing an efficient method that combines Haar-like features, integral images, and the AdaBoost algorithm. The haar-like features represent patterns in the image, such as edges, corners, and changes in intensity. The integral image technique is employed to enhance the speed of feature extraction by allowing rapid calculation of the sum of Haar-like features within any rectangular region of an image. The AdaBoost technique is applied to pick out a subset of the most distinguishing features and create a strong classifier by combining weak classifiers. The adaptive boosting technique focuses on misclassified samples in each iteration, leading to an accurate and robust face detection model. In terms of facemask detection, the Viola-Jones algorithm itself does not inherently include specific mask-detection capabilities. However, its underlying principles of object detection can be adapted for facemask detection by retraining the cascade of weak classifiers using masked and unmasked face data. Unlike Viola-Jones algorithm, CNN-based algorithms such as COCO, YOLO have demonstrated significant improvement in the task of detecting objects and have enhanced ability to train with a larger number of classes [18]. Based on this concept, this study adopts a CNN model together with the COCO model to augment its capability to precisely identify individuals wearing masks correctly, not wearing masks, and wearing masks improperly.

The subsequent sections of this paper are organized as follows: The Methods section delineates the methodology employed in this study. Following that, the Results and Discussion segment delineates the experiments conducted and deliberates on the evaluation outcomes. Lastly, the Conclusions section concludes the article and proposes potential avenues for future research.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

Figure 1. Examples of Correctly and Incorrectly Worn Facemask [17]

2. METHODS

The framework-based methodology [20] adopted for this research is presented in Figure 2 using the architecture of the proposed system. The facemask detection system aims to determine whether individuals are wearing masks appropriately, not wearing masks, or wearing them incorrectly by using real-time video streams from a webcam. The system is implemented using Python programming language and various libraries and frameworks such as COCO, OpenCV, Keras, and TensorFlow were utilized. The key components of the proposed system are described below.

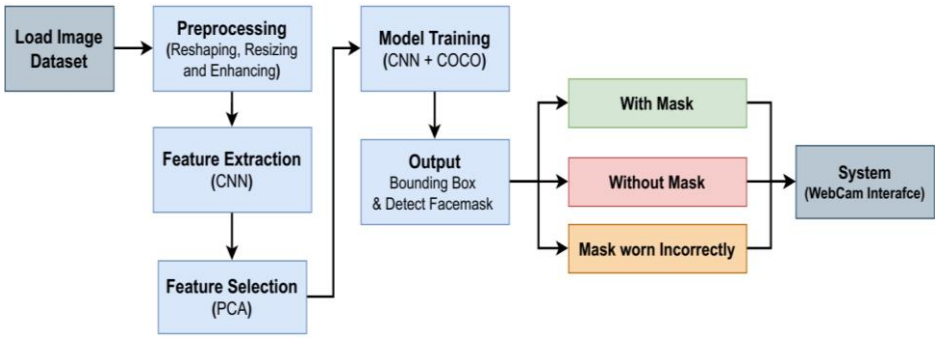

Figure 2. Architecture of Proposed System

2.1 Dataset Collection

The first stage of the facemask detection system is acquiring the dataset. After obtaining the images, various image processing methods can be applied. If the images are not acquired correctly, it becomes difficult to achieve the desired tasks. There are different methods for acquiring data depending on the type of research being done. In this research project, secondary data collection methods are adopted, which rely on previously provided data such as printed materials, textbooks, periodicals, journals, and other publications. Secondary data from online journals and articles are used in this research project. The dataset for the facemask detection task consists of 853 images depicting individuals wearing various types of masks, categorized as "with mask," "without mask," and "mask worn incorrectly," were included." The images are stored in three different folders.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

The dataset's size and distribution are rationalized to ensure that the trained machine-learning model can accurately detect individuals wearing masks in various situations. See Figure 3 for a snapshot of the dataset.

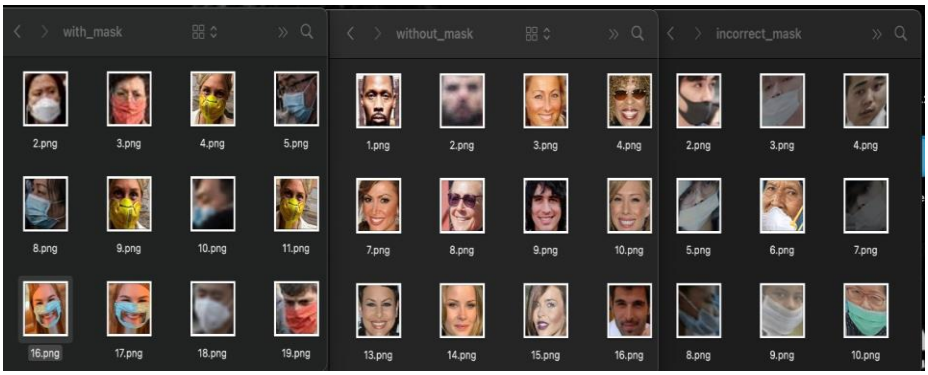

Figure 3. Snapshot of Dataset

2.2 Preprocessing

Images obtained from various sources may not have the same quality. It is suggested to use some image preprocessing techniques to maintain the features of photos and improve picture quality. One way to do this is by changing the intensity distribution to alter the contrast of a picture; in this case, constant restricted adaptive histogram equalization is used. This prepares the image for feeding into a CNN for feature extraction. During the preprocessing stage, the raw images from the dataset are transformed and enhanced to enhance the general efficiency of the machine learning of the proposed model. The preprocessing function "preprocess image (images, labels)" (see Figure 4) consists of several image processing techniques applied sequentially. Data labeling: After data augmentation, the next step is to label the images as "with mask," "without mask," or "incorrect mask." This labeling is essential for the CNN algorithm to learn to recognize faces wearing mask accurately.

1) Resizing: by pre-processing, the size of the pixels is resized to a standard image size representation.
2) Data normalization: Data normalization stands as a pivotal stage in data preprocessing, entailing the adjustment of pixel values to conform to a consistent range. This step ensures that the CNN algorithm can process the data efficiently.
3) Enhancing: converting the color format of the images to a readable format understood by the ML technique.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

4) Face alignment: After detecting faces, the faces should be aligned to a fixed size and orientation. This step ensures that the CNN algorithm can process the faces efficiently.

```
16  def preprocess_data(images, labels):
17      processed_images = []
18      for image in images:
19          image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
20          image = cv2.resize(image, (224, 224))
21          image = tf.keras.applications.mobilenet_v2.preprocess_input(image)
22          processed_images.append(image)
23      return np.array(processed_images), np.array(labels)
```

Figure 4. Python Code for Image Preprocessing

In figure 4, the function preprocess_data (images, labels) performs a series of sequential image_processing techniques. First, it creates and initializes an empty list called "processed_images". Then, it loops through each "image" in the "images" list. The next step involves color representation and resizing. It uses the cv2.cvtColor function to convert the color space of the image, and the cv2.resize function to resize the image to a specific size, typically required for compatibility with machine learning models. After processing the image, it appends the processed image to the "processed_images" list and returns the processed images and their corresponding labels.

2.3 Feature Extraction

When performing facemask detection, it is necessary to extract local features, such as the type and pattern of masks. CNN's approach involves extracting features at a high resolution to create a feature map. This is achieved through two fully connected layers and two part-one convolution layers. The core component of CNN, convolution, gathers intricate elements from images to form a feature map. The array is then flattened and sent to a fully connected layer for training, testing, and validation with the provided labels. Each image is divided into blocks or patches of the same size, also known as receptive fields. These blocks can be non-overlapping or overlapping. Overlapping blocks are considered to extract smooth features, as they share aspects of the image that non-overlapping blocks do not. See Algorithm 1 for steps involved in feature extraction.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

Algorithm 1. Feature Extraction Using CNN

Step 1: Begin (CNN)
Step 2: Load dataset
Step 3: Execute data preprocessing (resizing, normalization, and augmentation.
Step 4: Divide dataset into train and test set.
Step 5: Design the CNN architecture.
- Initialize sequential layer.
- Add these layers: convolution, pooling, fully connected.
- Output layer with an activation function.

Step 6: Assemble the model with 'Categorical_Crossentropy' loss function, 'Adam' optimizer, and 'accuracy' metric.
Step 7: Preprocess the dataset.
- Normalize the pixel values to be between 0 and 1.
- Split the dataset into training and validation sets.

Step 8: Train the CNN
Step 9: Extract features from the images.
- Remove the output layer of the CNN.
- Pass each image through the modified CNN to obtain a feature vector for each image.
- Store the feature vectors in a new dataset.

In Figure 5, a function called "extract_features" extracts features from regions of interest (ROIs) within an image based on bounding box coordinates. The function takes "image" and "bboxes" (a list of bounding boxes) as parameters and creates an empty list called "feature_list" to store the extracted features. It computes the bounding box coordinates and isolates the region of interest (ROI) from the image. The ROI is then resized to 224 x 224 pixels and converted from BGR to RGB color representation. The resulting feature vectors are then appended to the "feature_list". Finally, the function returns the complete list of feature vectors, which is a 2-dimensional list with the dimensions being the number of bounding boxes and the size of each feature vector.

```
26  # Define function for feature extraction
27  def extract_features(image, bboxes):
28      feature_list = []
29      (h, w, d) = image.shape
30      for bbox in bboxes:
31          xmin, ymin = int(bbox[1]*w), int(bbox[0]*h)
32          xmax, ymax = int(bbox[3]*w), int(bbox[2]*h)
33          roi = image[ymin:ymax, xmin:xmax]
34          roi = cv2.resize(roi, (224, 224))
35          roi = cv2.cvtColor(roi, cv2.COLOR_BGR2RGB)
36          feature = np.array(roi).flatten()
37          feature_list.append(feature)
38      return feature_list
```

Figure 5. Python Code for Feature Extraction using CNN

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

2.4 Feature Selection

When using a machine learning classifier to address the detection problem, processing high-dimensional feature vectors of images can be time-consuming and resource-intensive. The PCA method is used to categorize the features into three labels by selecting relevant features from the dataset that can effectively identify or classify objects. This technique represents original data as feature vectors and reduces overfitting, contributing to feature vector size reduction. The approach often results in a linear combination of the most variable elements and translates the image matrix into a lower-dimensional subspace through various operations. The covariance matrix is used to illustrate the relative variation between pixels in an image, and eigenvectors are subsequently determined from this matrix. Principal components are the eigenvectors with the highest eigenvalues. Images obtained from different sources may not be of the same quality, so some image preprocessing techniques are suggested to maintain the features of photos and improve picture quality. See Algorithm 2 for steps involved in feature selection.

Algorithm 2. Feature Selection Using PCA

Step 1: Input vectors of extracted image data
Step 2: Standardize the dataset.
- Calculate mean and the standard deviation of each feature.
- Subtract the mean from each sample and divide by the standard deviation.

Step 3: Compute the covariance matrix.
- Calculate the covariance between each pair of features.

Step 4: Compute the eigenvectors and eigenvalues.
- Calculate the eigenvectors and eigenvalues of the covariance matrix

Step 5: Select the top k eigenvectors.
- Arrange the eigenvalues in descending order.
- Choose the top k eigenvectors corresponding to the k largest eigenvalues.

Step 6: Project the dataset onto the selected eigenvectors. Multiply the standardized dataset by the selected eigenvectors to obtain the new feature space.
Step 7: Select the first k columns of the new feature space. Select the first k columns of the new feature space as the subset of k features.

In Figure 6, a function called "select_features" is defined to take a parameter “feature”. It creates an instance of the PCA class with "n_components" set to 50, representing the number of principal components to be retained after dimensionality reduction. The PCA model is fitted to the input features using the fit method, learning the statistical properties of the data and computing the principal components. The input attributes undergo a conversion into a unique feature domain delineated by the chosen principal components via the utilization of the transform function. The resulting selected_features array represents the transformed features in the new feature space.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

```
# Define function for feature selection using PCA
def select_features(features):
    pca = PCA(n_components=50)
    pca.fit(features)
    selected_features = pca.transform(features)
    return selected_features
```

Figure 6. Python code for Feature Selection using PCA.

2.5 Model Training

When designing a facemask detection system, a Convolutional Neural Network (CNN) undergoes training utilizing the COCO dataset to recognize various entities, including faces and facemasks. Throughout the training phase, the CNN is exposed to positive instances (images depicting individuals wearing facemasks), negative instances (images of individuals not wearing facemasks), and neutral instances (images of individuals wearing facemasks incorrectly). This training employs backpropagation, a method involving the retroactive propagation of errors through the network, and adjustment of network weights to minimize errors. The input image undergoes classification against the COCO dataset to predict its category, which could be a face with a mask, without a mask, or an improperly worn mask. The effectiveness of the trained model is assessed using evaluation metrics such as Intersection over Union (IoU) and Mean Average Precision (mAP) on an independent validation dataset that was not used during the training process. Refer to Algorithm 3 and 4 for a breakdown of the steps involved in model training.

The CNN model goes through several layers after uploading the dataset. Image classification involves extracting features from the image to identify patterns in the dataset. For each entry that is less than a 2x2 matrix, a neuron is produced and linked to all additional neurons in the following layer, resulting in fewer dimensions and less training time. Terms such as feature map, convolved feature are used to define the output of the convolution operation. A feature map is generated employing a feature detector, commonly represented as a 3x3 matrix. This feature map is generated through element-wise multiplication of the kernel with the input image, followed by aggregation of the outcomes. Subsequently, the fully connected layer assimilates insights from the derived features, discerning which elements are pivotal in determining whether an individual is wearing a facemask correctly, not wearing one, or wearing it incorrectly.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

Algorithm 3. COCO Model

Step 1: Load the COCO dataset and the predicted outputs from the model.
Step 2: For each image in the dataset, extract the ground truth annotations and the predicted outputs.
Step 3: Perform object detection and pass the pre-processed image(s) through the COCO model. The model outputs a set of bounding boxes and class probabilities for objects in the image(s)
Step 4: Postprocess the output.
- Apply non-maximum suppression to remove duplicate or overlapping detections.
- Filter out detections below a certain confidence threshold.
- Convert the bounding box coordinates from the model's output format to pixel coordinates in the original image(s).
- Map the class probabilities to human-readable class labels using the COCO dataset's mapping.

Step 5: Output the detected facemask. Return a list of the detected objects along with their class labels and bounding boxes.
Step 6: Calculate the precision and recall values for the predicted outputs.
Step 7: Calculate the average precision and recall values for each object category.
Step 8: Calculate the mean average precision (mAP) and the mean average recall (mAR) overall object categories.
Step 9: Output the mAP and mAR values as model evaluation metrics.

Algorithm 4. Combining CNN and COCO Model

Step 1: Load the dataset of images with facemasks and without facemasks and incorrect facemasks.
Step 2: Divide the dataset into train and test sets.
Step 3: Preprocess the images:
- Resize the images to a common size.
- Convert the images to grayscale and normalize the pixel values of the images.
- Apply image augmentation techniques to increase the size of the training set and to reduce overfitting.

Step 4: Define the CNN architecture with the following layers: Convolutional Layer, Max Pooling Layer, Dropout Layer, Flatten Layer, Dense Layer, Output Layer.
Step 5: Compile the model with a suitable loss function and optimizer.
Step 6: Train the model on the training set.
Step 7: Evaluate the performance of the model on the testing set.
Step 8: Use the model to detect facemasks in real-time video stream from webcam:
- Initialize the video stream from the webcam and Loop over the frames in the video stream.
- Extract the face from each frame using a face detection algorithm.
- Preprocess the extracted face using the same preprocessing steps.
- Pass the preprocessed face through the trained model to detect the presence of a facemask.
- Draw a bounding box around the detected face and label it as "with facemask", "without facemask" or "incorrect facemask".
- Display the resulting video stream with the detected facemasks.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

3. RESULTS AND DISCUSSION

3.1 Detection Outcomes

The face mask detection system in this study analyzes video frames to perform real-time detection of faces. The detection outcome includes a bounding box that indicates the location of the detected face object, its category, and the degree of probability (i.e. confidence level) If the system identifies that the subject is enclosed within the bounding box and recognizes a face, and if it concludes that the individual is wearing a mask properly, it labels the result as "With Mask. For example, in Figure 7, this detection is shown with a confidence level of 90%, indicating compliance with safety guidelines. If the system detects a face and concludes that the person is not wearing a mask, it categorizes the outcome as "Without Mask", as shown in Figure 8 with a confidence level of 99%. When the system detects a face with an object resembling a mask but determines that the mask is not worn correctly, the outcome is classified as "Mask Worn Incorrectly", as illustrated in Figure 9 with a confidence level of 91%.

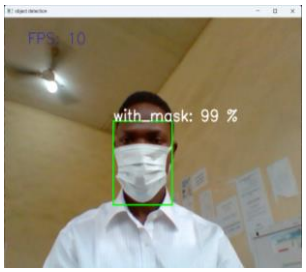

Figure 7. Detected Face With Mask

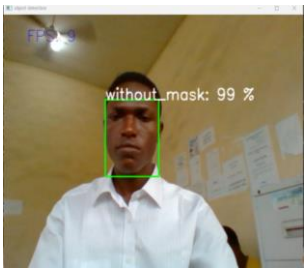

Figure 8. Detected Face Without Mask

Figure 9. Detected Face With Mask Worn Incorrectly

Thus, the bounding box serves as a reference area for facemask detection within a digital image, represented as an imaginary rectangle overlaid on the image, and is also used in evaluating the performance of the detection model.

3.2 Detection Model Evaluation Metrics

Accurate evaluation metrics are essential for assessing the performance of any object detection model, such as our COCO model designed for facemask detection [18]. The predominant measure utilized for evaluating the precision of detection models is Average Precision (AP). Another significant metric is Intersection over Union (IoU), pivotal for evaluating the detection model by defining "correct detection" and "incorrect detection". IoU is integral in AP computation and assesses the model's capability to accurately pinpoint objects within images by determining the ratio of overlap area between the predicted

bounding box and the ground truth bounding box to the union area of the two bounding boxes. Mathematically, IoU is represented in Equation 1.

$$IoU = \frac{area\ of\ overlap}{area\ of\ union} \quad (1)$$

The Intersection over Union (IoU) serves as a mathematical representation that quantifies the alignment between a model's prediction and the genuine object position, offering a numerical assessment. IoU values range between 0 and 1, where 0 denotes no overlap, and 1 signifies a flawless alignment between the predicted and actual bounding boxes. By juxtaposing IoU against a specified threshold "k", we can categorize a detection as either accurate or inaccurate. When IoU equals or exceeds "k", the detection is deemed accurate, while IoU less than "k" indicates an inaccurate detection.

Average Precision (AP) stands as a measure employed to assess detection models, evaluating precision-recall values across diverse IoU (Intersection over Union) thresholds. Precision denotes the accuracy of positive predictions made by the model, reflecting the proportion of accurate positive predictions. Meanwhile, recall evaluates the model's proficiency in identifying all relevant objects, indicating the ratio of correct positive predictions relative to all true objects. The average precision (AP) is determined by plotting the precision-recall curve and computing the area under this curve. This curve illustrates the equilibrium between precision and recall across various confidence levels associated with bounding boxes generated by a detection model.

Mean Average Precision (mAP) extends further the concept of AP by computing the average AP across multiple IoU thresholds to assess the model's performance. Mathematically, mAP is expressed in Equation 2.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \quad (2)$$

where: N is the number of object categories and APi is the value of AP for category i.

Average Recall (AR) and Mean Average Recall (mAR) are two crucial metrics for evaluating detection models. Mean Average Recall assesses the models' capability to detect objects of various sizes and at different confidence levels. Mathematically, mAR is expressed as follows:

$$mAR = \frac{1}{N} \sum_{i=1}^{N} AR_i \quad (3)$$

where: *N* is the number of object categories and AR*i* is the value of AR for category *i*.

3.3 Detection Model Evaluation

3.3.1 Evaluation Results 1

This research centered on facemask detection using a hybrid machine learning-based model. The detection model was trained with CNN and COCO to detect faces with masks, without a mask and incorrectly worn masks. Figure 10 presents the results of our facemask detection model evaluation using Average Precision and Average Recall metrics, both calculated using intersection over union (IoU) and area thresholds. There are three commonly used IoU thresholds (these include 0.50, 0.50-0.95, and 0.75) but the metrics used in work were computed over the range of 0.50-0.95 thresholds. Additionally, AP for small, medium, and large objects is calculated separately to assess the model's performance for different object sizes. The objects are categorized into these sizes based on the dimensions of their bounding box or area.

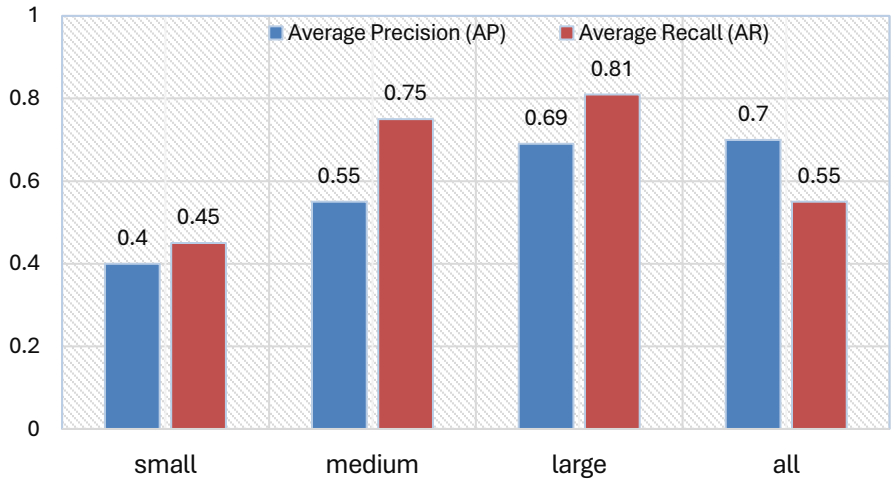

Figure 10. Results of Facemask Detection Model Evaluation

The reported AP value of 0.70 in Figure 10 means that the model achieved relatively high accuracy in localizing objects within the images. It is calculated as the average recall at diverse levels of precision, which is the portion of pertinent cases that are recovered by the model. The reported AR value of 0.81 in Figure 10 indicates that the model identified a high percentage of all instances of the object in the images. Both metrics are evaluated on objects of "large" area, meaning that the model is primarily evaluated on objects that occupy a sizeable portion of the

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

image. This recommends that the model implements well in identifying and localizing larger objects within the images, but its performance may be lower for smaller objects. This result indicates that the facemask detection model has relatively high accuracy in identifying and localizing objects of all sizes within images.

Thus, the average precision (AP) and average recall (AR) at 50- to-95 thresholds for small, medium, and large areas were assessed to understand how effective the facemask detection model is for various object sizes and levels of overlap. This evaluation method allows us to gain a comprehensive knowledge of the advantages and constraints of the model in various situations.

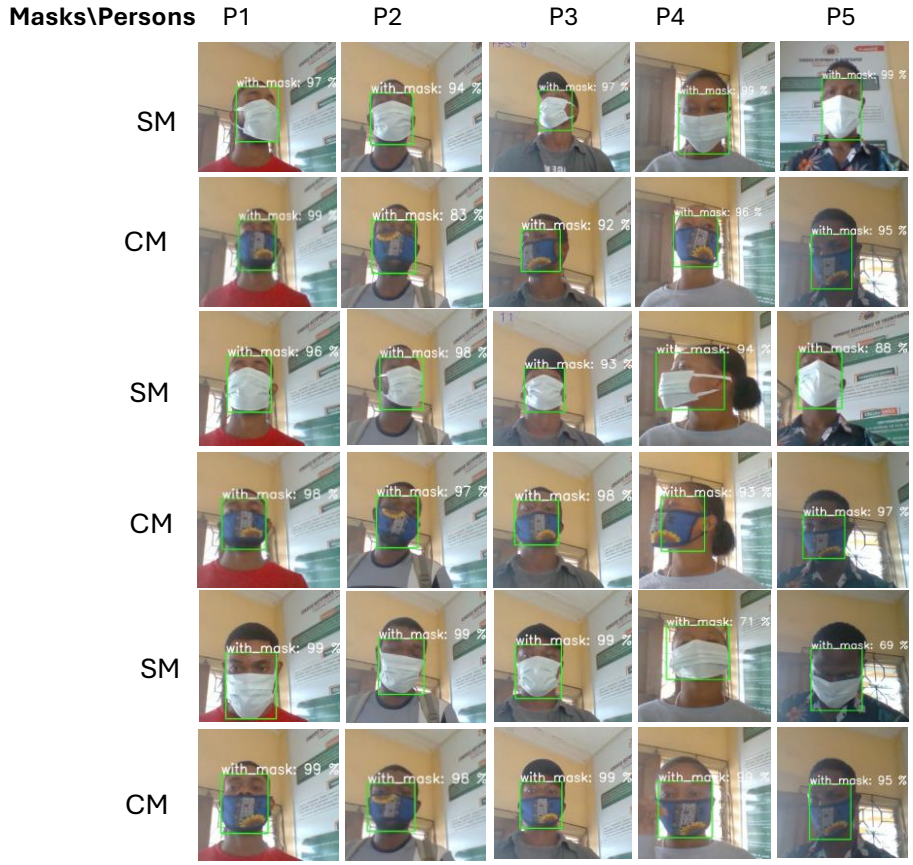

Figure 11. Detected faces with different mask types

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

3.3.2 Evaluation Results 2

The evaluation of the facemask detection model was expanded to consider different individuals and various types of facemasks. The goal was to demonstrate the variability of the model's performance when detecting people wearing facemask in different scenarios. For simplicity, we only considered two types of facemasks: cloth masks (labeled as CM) and surgical masks (labeled as SM). Additionally, we selected five individuals (labeled as P1, P2, P3, P4, and P5) for the evaluation, and each person wore both types of facemasks for detection by the model. This resulted a total of 30 detection scenarios, as shown in Figure 11.

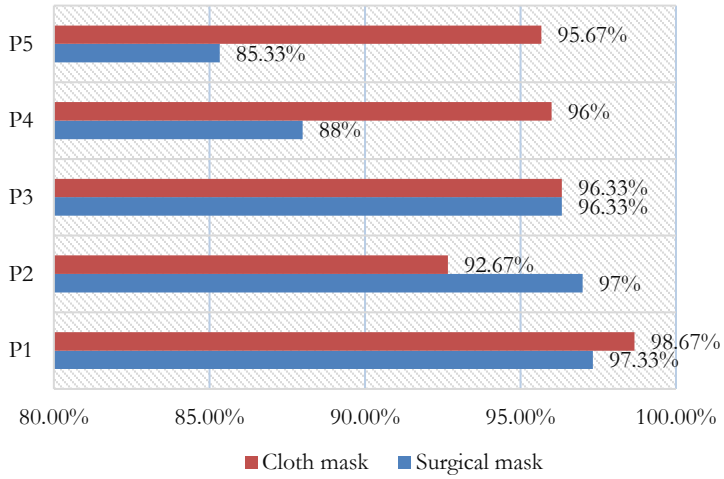

Figure 12. Mean detection accuracy per person based on the mask type worn.

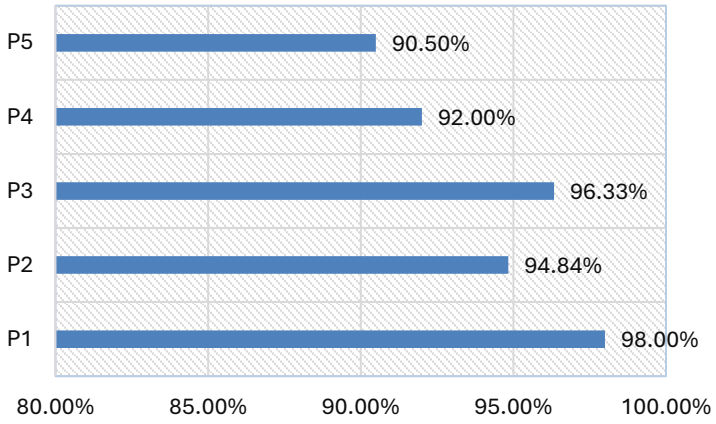

Figure 13. Mean detection accuracy per person.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

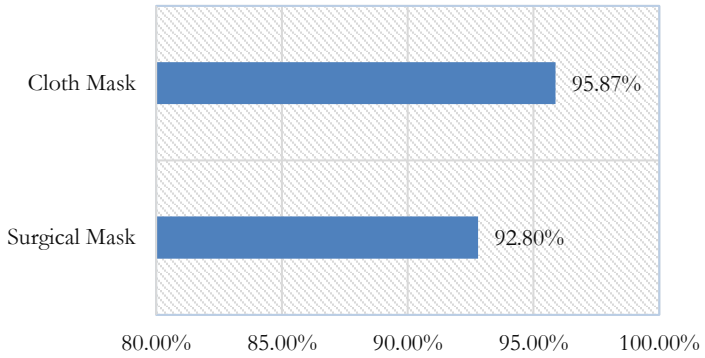

Figure 14. Mean detection accuracy per mask type worn.

In Figures 12, 13, and 14, the model's performance evaluation is presented in terms of mean detection accuracy computed using confidence levels. Overall, the mean accuracy varies in the detection scenarios involving different persons wearing the same type of facemasks and in scenarios involving the same person wearing different types of facemasks. Specifically, Figure 13 shows the mean accuracy from detecting persons with cloth masks and surgical masks. It indicates that the detection scenario involving a person labeled as P1 resulted in the highest mean accuracy for both types of masks, with cloth masks achieving above 98% accuracy and surgical masks falling below 98%. Figure 13 presents the computed mean accuracy from detecting each person across all detection scenarios. It demonstrates that the scenarios involving a person labeled as P1 resulted in the highest mean accuracy (above 98%), while those involving a person labeled as P5 resulted in the lowest mean accuracy (below 98%). In Figure 14, the computed mean accuracy from detecting the different types of facemasks is presented. It shows that cloth mask detection had a higher mean accuracy value (above 95%) compared to the detection of surgical masks (below 95%)

4. CONCLUSIONS

This research focused on developing a facemask detection system using a combination of CNN, PCA and COCO. The study highlighted the necessity of accurate facemask detection amidst the COVID-19 pandemic to mitigate virus transmission through respiratory droplets. Different faces and facemask types, including surgical masks and reusable cloth masks, were considered to ensure the detection system's applicability in diverse real-world scenarios. Nevertheless, key findings include the effectiveness of CNN for feature extraction and PCA for feature selection, which collectively enhanced the model's accuracy and efficiency. The integration of COCO datasets facilitated robust training, leading to reliable detection of correctly worn, incorrectly worn, and absent facemasks. Evaluation metrics such as AP and AR were used to assess the model's performance,

demonstrating high accuracy in detecting facemasks under various conditions. The results in this study underscore the potential of hybrid models in improving detection capabilities, especially in real-time applications. Future research could further optimize these methods for even greater accuracy and broader applicability. The developed system presents a significant step forward in utilizing machine learning for public health safety, offering a reliable tool for ensuring compliance with facemask guidelines in public and private settings.

REFERENCES

[1] World Health Organization, "Coronavirus disease 2019 (COVID-19): situation report, 73," 2020.

[2] C. G. Dwirusman, "The Role and Effectivity of Face Mask in Preventing Transmission of Coronavirus Disease 2019 (COVID-19)," *Jurnal Medika Hutama*, vol. 2, no. 01, pp. 412-420, Oct. 2020.

[3] D. Kumar, R. Malviya, and P. K. Sharma, "Corona virus: a review of COVID-19," *EJMO*, vol. 4, no. 1, pp. 8-25, 2020.

[4] A. Nowrin, S. Afroz, M. S. Rahman, I. Mahmud, and Y. Z. Cho, "Comprehensive review on facemask detection techniques in the context of covid-19," *IEEE Access*, vol. 9, pp. 106839-106864, 2021.

[5] A. Nieto-Rodriguez, M. Mucientes, and V. M. Brea, "System for medical mask detection in the operating room through facial attributes," in *Pattern Recognition and Image Analysis: 7th Iberian Conference, IbPRIA 2015, Santiago de Compostela, Spain, June 17-19, 2015, Proceedings 7*, Springer International Publishing, pp. 138-145.

[6] M. S. Ejaz, M. R. Islam, M. Sifatullah, and A. Sarker, "Implementation of principal component analysis on masked and non-masked face recognition," in *2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, 2019, pp. 1-5.

[7] B. Qin and D. Li, "Identifying facemask-wearing condition using image super-resolution with classification network to prevent COVID-19," *Sensors*, vol. 20, no. 18, p. 5236, 2020.

[8] M. Jiang and X. Fan, "Retinamask: A face mask detector," *arXiv:2005.03950*, 2020.

[9] B. Batagelj, P. Peer, V. Štruc, and S. Dobrišek, "How to correctly detect face-masks for covid-19 from visual information?," *Applied Sciences*, vol. 11, no. 5, p. 2070, 2021.

[10] M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa, "Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection," *Sustainable Cities and Society*, vol. 65, p. 102600, 2021.

[11] P. Nagrath et al., "SSDMNV2: A real time DNN-based face mask detection system using single shot multibox detector and MobileNetV2," *Sustainable Cities and Society*, vol. 66, p. 102692, 2021.
[12] N. C. Ristea and R. T. Ionescu, "Are you wearing a mask? Improving mask detection from speech using augmentation by cycle-consistent GANs," *arXiv:2006.10147*, 2020.
[13] M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa, "A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic," *Measurement*, vol. 167, p. 108288, 2021.
[14] H. M. Al-Sarrar and H. H. Al-Baity, "A novel hybrid face mask detection approach using Transformer and convolutional neural network models," *PeerJ Computer Science*, vol. 9, p. e1265, 2023.
[15] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, pp. I-511, 2001.
[16] T. Y. Lin et al., "Microsoft COCO: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer International Publishing, pp. 740-755.
[17] Olmsted Medical Center, "Wearing A Mask," 2024. [Online]. Available: https://www.olmmed.org/covid-19-information/how-to-wear-a-mask/.
[18] R. Padilla, S. L. Netto, and E. A. Da Silva, "A survey on performance metrics for object-detection algorithms," in *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2020, pp. 237-242.
[19] I. Umoren, S. Inyang, & A. Silas, "Intelligent Surveillance and Facial Recognition System for Efficient Border Monitoring and Threats Prediction Using Machine Learning Approach," *Researchers Journal of Science and Technology*, vol. 1, no. 1, pp. 47-63.
[20] I. Omoronyia, U. Etuk, and P. Inglis, "A privacy awareness system for software design," *International Journal of Software Engineering and Knowledge Engineering*, vol. 29, no. 10, pp. 1557-1604, 2019.