Mobile Recognition of Image Components Based on Machine Learning Methods

Galyna Kondratenko[1], Ievgen Sidenko[1,*], Maksym Saliutin[1] and Yuriy Kondratenko[1,2]

[1]*Petro Mohyla Black Sea National University, 68th Desantnykiv Str., 10, Mykolaiv, 54003, Ukraine*
[2]*Institute of Artificial Intelligence Problems, Mala Zhytomyrs'ka Str., 11/5, Kyiv, 01001, Ukraine*
E-mail: halyna.kondratenko@chmnu.edu.ua; ievgen.sidenko@chmnu.edu.ua; salyutin1@gmail.com; yuriy.kondratenko@chmnu.edu.ua
[*]*Corresponding Author*

Received 06 November 2023; Accepted 04 March 2024

Abstract

This paper is related to the recognition of certain components in images using machine learning methods and mobile technologies. The main result of this work is a developed system for recognizing the presence of a mask on the face using an image, which provides all the necessary information in real-time about the presence or absence of a mask on the face. When the program is turned off, statistics about the presence/absence of the mask will be recorded in the database. To achieve the goal, the following tasks were solved: the current state of the task of recognizing the presence of a mask on a person's face was analysed; existing analogs of the systems were analysed; the necessary neural network architecture was selected as one of the machine learning methods; developed a system for recognizing the presence of a mask on the face using the necessary libraries; a user graphical interface, a database model for recording statistics and additional functionality have been developed; conduct testing. Practical application has a fairly wide range, in particular, the developed intelligent system is intended for use in the subway, industrial enterprises, state institutions, educational institutions, offices, and other public places. The developed system recognizes and records statistics about the presence of a mask on a person's face using neural networks.

Keywords: Mobile recognition, image components, mask, machine learning methods, neural networks.

1 Introduction

One of the urgent tasks of image recognition is recognizing the presence of a mask on a person's face, taking into account the coronavirus disease (COVID-19) pandemic and its possible consequences, as well as the mask mode at various enterprises that work with harmful substances [1–4]. Following the rapid spread of a new case of COVID-19, the World Health Organization (WHO) has confirmed that it is a dangerous virus that can be transmitted from person to person through airborne droplets [1]. In terms of prevention, everyone should wear a face mask, practice social distancing, avoid crowded places, and keep your immune system up at all times. Therefore, to protect each other, each person should properly wear a protective medical mask when in a public gathering. In addition, wearing a suitable mask on the face when working with dangerous harmful substances at enterprises is also mandatory for the protection of one's own health and that of others [5–7]. This is due to the fact that a situation may arise when a person without a mask may lose consciousness when working with harmful substances (for example, by inhaling vapors) and thus endanger others if this substance spills around. However, some irresponsible people refuse to wear a mask despite all the dangers and consequences. Therefore, the development of a system, in particular using mobile technologies, to recognize the presence of a face mask in this case is very important [8–10]. Manually tracking the presence of a mask on people's faces is a difficult task and requires a lot of effort and additional staff. Therefore, it is urgent to create a system that will monitor the presence of a mask, make statistics, which will generally help people to save their lives.

The main research objectives are the methods and technologies of mask recognition on a person's face and the development of an application for the effective implementation of this process. Using mobile phones for mask recognition can make the process fast, convenient, and massive [4]. This is especially true in situations where many people need to be screened, such as at large events or in public places. Therefore, it is important to create a system in the form of an application for PC and Android versions that will track the presence of a mask on a person's face when entering an enterprise or in public places, keep statistics on the number of people with and without a mask, which will generally help people save their lives and ensure safety measures.

2 Related Works and Problem Statement

In this difficult time, many companies and corporations have been able to properly adapt and develop whole systems to preserve people's health and ensure their safety. Scientists and developers have created automated systems for recognizing masks in public places and places with an increased risk of poisoning by vapors from dangerous harmful substances. Other researchers have also developed their own techniques for tracking face masks in public spaces [4, 6, 7]. Even in new versions of Apple phones, you can use the Face ID function without removing the medical mask from your face.

Also, now, mask recognition systems are gradually appearing in places where people gather to gain access to a public institutions.

Let's consider some analogues of systems for recognizing masks on a person's face. Apple iPad 8 tablet model with the MaskCheck application installed on a special mount (https://getmaskcheck.com/). This system is placed at the entrance to gain access to a public institution or, for example, the premises of an enterprise that works with harmful substances. The advantages are (a) instant messages about the presence or absence of a mask on a person's face, (b) easy to install and configure the app on your tablet. Limitations are (a) the high price for the system due to the additional purchase of the iPad 8, (b) no statistics collection per day, (c) requires constant connection to the WI-FI network, (d) lack of privacy, due to closed access to the program, (e) there is a risk of unauthorized access to information by the developer.

An application (https://safer.work/safe-space/) that can detect human faces in masks using a system based on machine learning. For installation, you can buy any tablet or phone with a front camera and install this device with the application in a public place. Advantages are (a) free application, (b) provides customized screen messages and sound alerts, (c) high accuracy of face mask detection. Limitations are (a) no statistics collection, (b) no privacy, (c) only for Android and iOS devices.

A program (https://felenasoft.com/) for video surveillance and any tasks of working with cameras. This system includes a large set of various cameras that can be installed in the rooms required for monitoring [11]. The advantages are (a) high accuracy of mask recognition, (b) does not require a constant Internet connection. The limitations are the need to buy a monthly subscription for constant maintenance and work with a cloud service to save statistics, (b) there is no sound accompaniment if the person does not have a mask on his face.

This paper [12] investigates the problem of the effect of wearing masks on the accuracy of automated facial recognition, which is important for contactless identification of people in various fields, such as border control, surveillance in public places, tracking attendance in schools, and registration of working hours.

This paper [13] considers the problem of face recognition in the presence of masks. Existing methods have problems with facial recognition when the nose and mouth are covered by a mask. Usually, the methods lose information from closed areas or try to recover these areas before recognition.

The authors [14] proposed a Quadruplet Loss model that generates closed vectors for both masked and unmasked photos. This model is based on triplet loss and includes synthesized masked images, forming a quadruple.

This paper [15] proposes a method to reduce the impact of mask defects on face recognition. An accurate and affordable masked face synthesis method is used, as well as the AMaskNet model to improve masked face recognition.

This work [16] uses a modified version of the deep neural network VGG19, which was trained on images with and without masks of 62 people. With this approach, it was possible to achieve an accuracy of 80 to 85 percent in face recognition on test images with a mask and different poses.

So, after reviewing all the above systems, you can highlight all the advantages and disadvantages. Among the advantages, the following can be highlighted: a high percentage of recognition and autonomy of systems. Among the limitations, it is possible to single out the lack of universal functionality of each considered system (multi-platform, keeping statistics, notifications, etc.), which makes it difficult to choose an affordable system to ensure the safety of citizens in crowded places and at enterprises with harmful substances. In previous works, similar Android applications and similar systems had limited functionality and did not provide for the use of different models of neural networks, they were more like a demo version that was not compatible with other systems in any way (for example, Telegram bot, where you can collect all statistics or use of the program itself on various operating systems: Android, Windows, Linux, etc.).

To address potential privacy concerns, particularly in public places where individuals are being monitored, it was suggested that visitors be notified of video surveillance to identify people without masks, which would generally reduce the risk of disease transmission. In addition, the video is not directly stored or transmitted to third parties.

Thus, the purpose of this work is to automate the process of recognizing the presence of a mask on a person's face and conducting statistics using neural networks through the development of a suitable multi-platform system.

3 Technologies for Object Recognition in Images

Currently, there are many neural network (NN) architectures. Each is suitable for a large number of tasks, but there is a large difference between them in the speed of learning, execution and adaptation to different platforms, as well as performance on weaker hardware. Therefore, the developer needs to study all the details of each architecture in order to choose the best one for his task [1, 5, 9, 17–25].

Considering the large number of studies and tests among researchers and developers, the convolutional neural network (CNN) showed the best results when solving the problem of object recognition in images or videos [1, 9, 22, 23, 26].

Nowadays, NNs already work directly on mobile phones with low-performance requirements, the AI built into the camera analyzes the environment and adapts the camera settings for better photo quality and identifies objects in seconds. Therefore, MobileNetV2 technology (according to Figure 1), which uses a modern convolutional network architecture, was used to solve the problem of recognizing a mask on a human face [27–29].

This architecture was chosen for the task at hand for several reasons. The basic building block of this network is generally similar to the previous generation but has several key features. As in MobileNetV1, there are convolutional blocks with a step of 1. Blocks with a step of 2 are designed to reduce the spatial dimension of the tensor and, unlike the block with a step of 1, do not have residual connections. Let's compare several network architectures. Let's take the well-known deep VGG16, and several variations of MobileNet (according to Table 1) [27–29].

The "Top-1 Accuracy" metric in Table 1 shows how many times the network predicted the correct label with the highest probability, and the

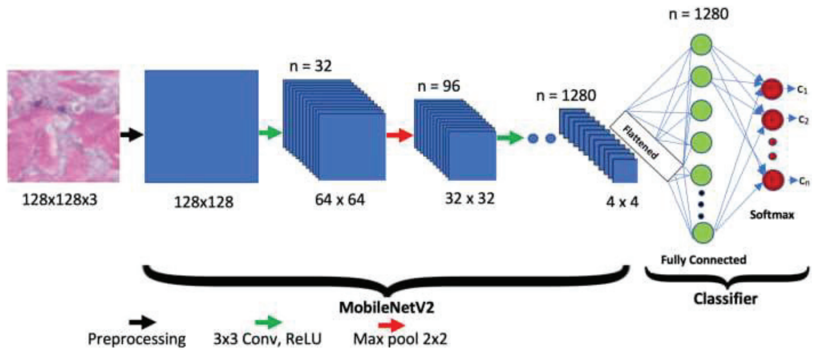


Figure 1 Architecture of MobileNetV2 network [27].

Table 1 Practical results of using neural network architectures

Network Architecture	Number of Parameters	Top-1 Accuracy	Top-5 Accuracy
VGG16	138.35M	0.715	0.901
MobileNetV1 (alpha = 1, rho = 1)	4.20M	0.709	0.899
MobileNetV1 (alpha = 0.75, rho = 0.85)	2.59M	0.672	0.873
MobileNetV1 (alpha = 0.25, rho = 0.57)	0.47M	0.415	0.663
MobileNetV2 (alpha = 1.4, rho = 1)	**6.06M**	**0.750**	**0.925**
MobileNetV2 (alpha = 1, rho = 1)	3.47M	0.718	0.910
MobileNetV2 (alpha = 0.35, rho = 0.43)	1.66M	0.455	0.704

"Top-5 Accuracy" metric shows how many times the correct label appears in the top five predicted classes.

The biggest achievement from these results is that there are network architectures that are capable of running on mobile devices and still show higher accuracy than VGG16. Using this MobileNetV2 architecture will allow you to create an optimized model that maintains a balance between accuracy and performance. MobileNetV2 has a limited resolution, which may limit the model's ability to recognize fine details or objects with a high degree of detail, but in the context of mask recognition, this does not greatly affect the system's performance. However, one of the main advantages of MobileNetV2 is its high speed [28].

The Python programming language has convenient functionality and the ability to use modern technologies that speed up the software development process. A ready-made dataset (https://www.kaggle.com/datasets/andrewmvd/face-mask-detection) consisting of 853 images of people, which are divided into three classes, namely "with a mask", "without a mask" and

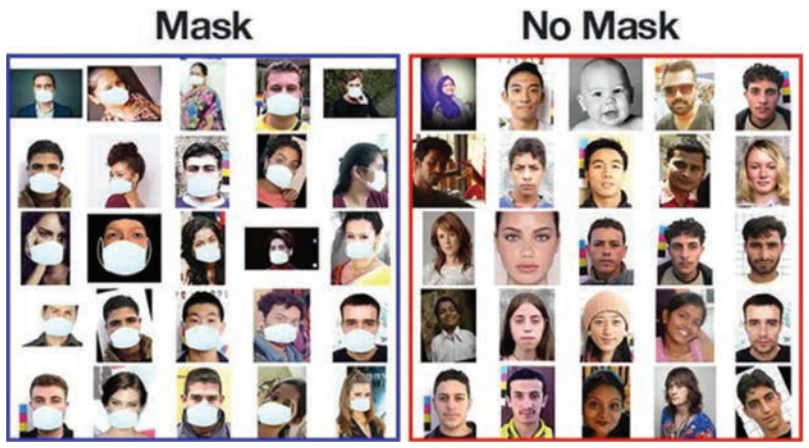


Figure 2 Data set for NN training.

"mask is worn incorrectly". It was decided to abandon the class "incorrectly worn mask" because the NN will recognize an incorrectly worn mask on the face as absent or "without a mask". Therefore, this data set was supplemented with images of people in masks and without masks and additionally annotated according to the PASCAL VOC format. The augmented dataset consists of 1920 images, divided into two classes required for future training (according to Figure 2): (a) with_mask – 966 images of people with a mask on their face, (b) without_mask – 954 images of people without a mask on their face.

For better training, the data set was sorted and distributed as follows: 80% for neural network training; 10% of the data set is allocated for the validation of the trained network; 10% for testing the final trained model.

Let's consider the learning process of a neural network in more detail.

Stage 1. Data download and preparation. At this point, the developed code uses the Keras library to load and preprocess images. The images are resized to (224, 224), which is the default size for MobileNetV2. Next, the data is divided into classes (mask or without mask). Then one-time encoding of labels and their conversion into one-hot encoding format is carried out. Finally, the data is divided into training, validation and test sets.

Stage 2. Data augmentation. This step uses the ImageDataGenerator to generate additional images using various transformations such as rotation, scaling, shifting, width and height displacement, mirroring, etc. This helps to expand the training set and reduce overtraining.

Stage 3. Building the model. At this stage, MobileNetV2 is used as the base model without upper fully connected (FC) layers. Next, an upper FC layer is added for class recognition (masked or not). At the same time, the base layers are frozen so that their weights are not updated during training.

Stage 4. Model compilation and training. At this stage, the Adam optimizer is used with a gradual decrease in the learning rate (learning rate decay). A binary cross-entropy loss function is also used since we have two classes (masked or not). The metric for determining efficiency is accuracy. The model is trained on the training data using the generated additional images.

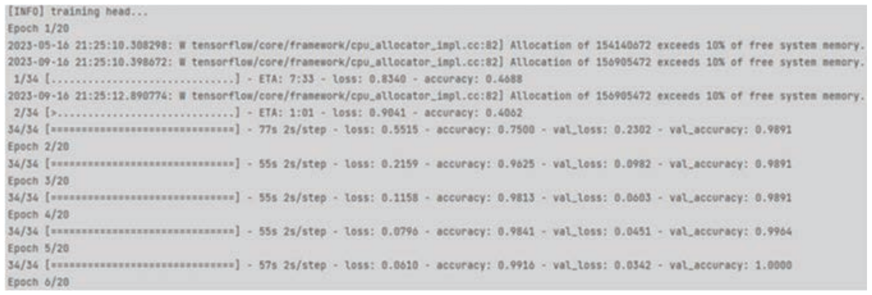
Stage 5. Evaluation and preservation of the model. After training, the model is evaluated on test data and a classification report is generated. The model is saved in H5 format for future use.

The default data augmentation hyperparameter settings were changed during training. The augmentation hyperparameters were changed to adapt to the specific properties of the task, namely: rotation_range = 30, which will help the neural network learn faces from different angles, zoom_range = 0.2, which will zoom in or out by 20%, which will help the model recognize objects of different sizes, shear_range = 0.2 changed to recognize faces in different angles.

These hyperparameters make it possible to reduce overtraining, improve the generalization of abstract and generalizing features, which can improve its ability to classify new images.

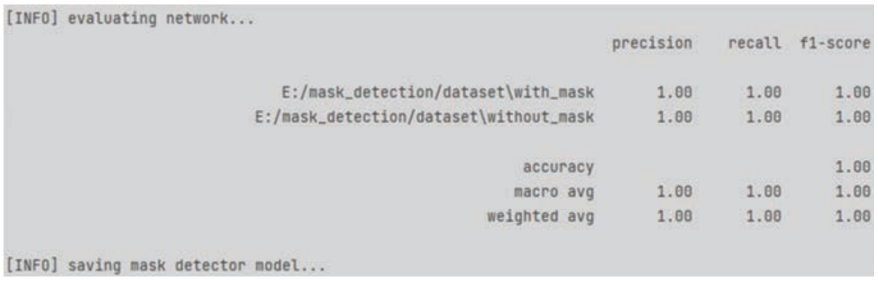
After running the code, the data collection and the learning process of the CNN begins (according to Figure 3).

In Figure 3, the following information can be seen: (a) number of epochs, (b) time spent per epoch, (c) number of learning losses (valued from 0 to 1), (d) accuracy (valued from 0 to 1). After all epochs, learning statistics are



```
[INFO] training head...
Epoch 1/20
2023-05-16 21:25:10.308298: W tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 154140672 exceeds 10% of free system memory.
2023-09-16 21:25:10.398672: W tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 156905472 exceeds 10% of free system memory.
 1/34 [..............................] - ETA: 7:33 - loss: 0.8340 - accuracy: 0.4688
2023-09-16 21:25:12.890774: W tensorflow/core/framework/cpu_allocator_impl.cc:82] Allocation of 156905472 exceeds 10% of free system memory.
 2/34 [>.............................] - ETA: 1:01 - loss: 0.9041 - accuracy: 0.4062
34/34 [==============================] - 77s 2s/step - loss: 0.5515 - accuracy: 0.7500 - val_loss: 0.2302 - val_accuracy: 0.9891
Epoch 2/20
34/34 [==============================] - 55s 2s/step - loss: 0.2159 - accuracy: 0.9625 - val_loss: 0.0982 - val_accuracy: 0.9891
Epoch 3/20
34/34 [==============================] - 55s 2s/step - loss: 0.1158 - accuracy: 0.9813 - val_loss: 0.0603 - val_accuracy: 0.9891
Epoch 4/20
34/34 [==============================] - 55s 2s/step - loss: 0.0796 - accuracy: 0.9841 - val_loss: 0.0451 - val_accuracy: 0.9964
Epoch 5/20
34/34 [==============================] - 57s 2s/step - loss: 0.0610 - accuracy: 0.9916 - val_loss: 0.0342 - val_accuracy: 1.0000
Epoch 6/20
```

Figure 3 The process of learning CNN on the MobileNetV2 architecture.



```
[INFO] evaluating network...
                                                   precision    recall  f1-score

                  E:/mask_detection/dataset\with_mask       1.00      1.00      1.00
               E:/mask_detection/dataset\without_mask       1.00      1.00      1.00

                                            accuracy                            1.00
                                           macro avg       1.00      1.00      1.00
                                        weighted avg       1.00      1.00      1.00

[INFO] saving mask detector model...
```

Figure 4 The result of the trained model.

displayed (according to Figure 4): (a) precision, (b) recall is completeness, namely how the CNN finds all the correct answers, (c) f1-score is interpreted as a harmonic mean between precision and image recall.

The F-score method is used to assess the quality of forecasting. There are several end results in the mask recognition problem. For this, the following designations will be considered [26]: (a) "True Positive", when a mask was detected, (b) "True Negative", when the network began to accept a hand covering a person's mouth and nose as a mask, (c) "False Positive, when the network identified any other object as a mask", (d) "False Negative", when the network failed to recognize an object on the human face, but it was a mask, (e) "Precision" is how well, the NN finds the object in the input image with minimal loss, (e) "Recall" is how correctly the NN determines the mask on the human face.

The goal of NN training is to minimize the false negative decision that degrades recognition accuracy [26, 30–32]. The "Precision", "Recall" and "F1-score" values are calculated according to the following formulas:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}, \tag{1}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}, \tag{2}$$

$$F1\text{-}score = \frac{2 \cdot Recall \cdot Precision}{Recall + Precision}. \tag{3}$$

The "Precision" metric indicates the accuracy of the recognized object in percent. The baseline metric used to evaluate a model is often Accuracy, which describes the number of correct predictions among all predictions. The "Recall" is a measure of how many positive predictions made are correct (true positives). A model with high recall accurately captures data associated with positive class labels and avoids false negative predictions. Models with high recall scores are necessary when the cost of running false negatives is high. The "F1-Score" is a measure that combines precision and recall. This is usually described as the harmonic mean of the two. The harmonic mean is another way of calculating the "average" of values, which is generally described as more suitable for ratios (such as precision and recall) than the traditional arithmetic mean [18–20].

To train the NN to recognize masks on a person's face from the total dataset, 20% of the data were selected for validation after successful completion of training, and the remaining 80% were used for training the NN. This distribution approach is better than manual shaping, because it automatically separates a specified percentage of data for validation, the only drawback is that in the case of training observations at specific moments, information about the images that were selected for validation can be useful [1, 26, 31].

The input layer receives data, in particular, an image with the extension ".jpeg" of 248x248 pixels. For this purpose, images are processed in the script before training, and then submitted to the NN input. If the size of the input data is large, then the calculations will require more time and resources, respectively. But if you reduce the photo to a small size, the NN will not be able to determine the features of the mask on the person's face and separate it as a key element. When fed to the input, the image is divided into 3 RBG channels (red, blue, green). The input layer normalizes the input data of each pixel into a certain range (from 0 to 1) according to the following formula [26, 31]:

$$f(p, \min, \max) = \frac{p - \min}{\max - \min}, \tag{4}$$

where $f(p, \min, \max)$ is the normalization function, p is the pixel color from 0 to 255, $\min$ is the minimum pixel value (ie 0), $\max$ is the maximum pixel value (i.e. 255).

The size of the convolutional layer can be calculated using the following formula [26, 33]:

$$(w, h) = (mW - kW + 1, mH - kH + 1), \tag{5}$$

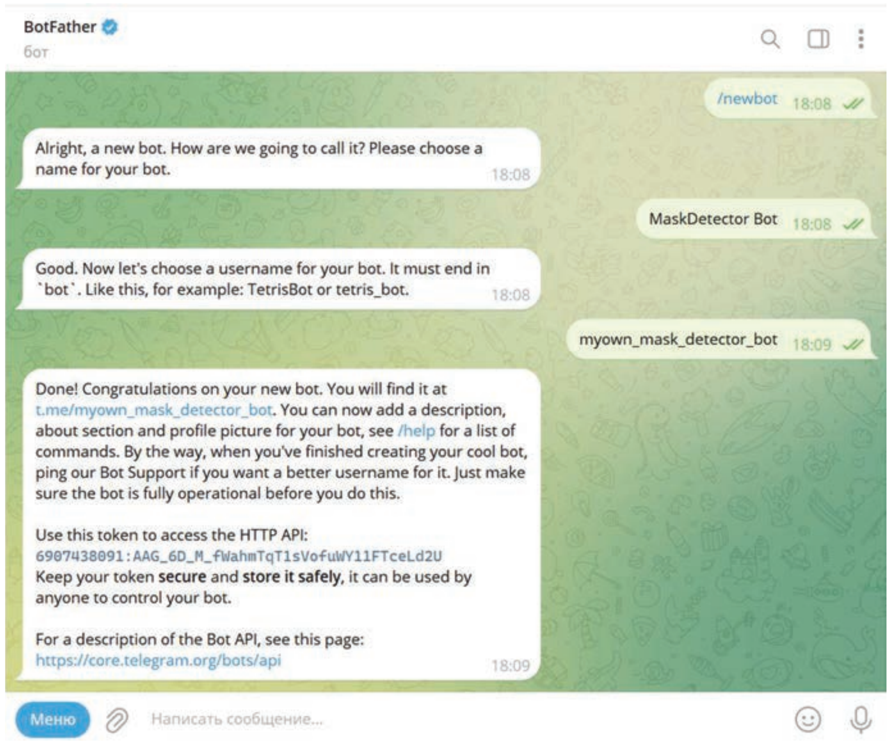
where (w, h) is the size of the convolution (feature) map, mW is the width of the previous map, mH is the height of the previous map, kW is the kernel width, kH is the kernel height.

The output layer of the CNN has a connection with all the neurons of the previous layer, the number of neurons at the output corresponds to the number of classes that were predetermined, that is, 2: (1) a person with a mask and (2) a person without a mask.

4 The Software of the Developed System and the Results of Modeling and Testing

When creating a system, the end user may need to collect information and generate the necessary statistics to understand the whole situation and how to proceed. Relational databases can be used for storage. These databases allow you to asynchronously record a large amount of information, code, modify and import it into other environments, as well as connect them to systems during the development process. SQLite is known for its ease of use and compactness. This property is important when working with a large amount of data, as it provides quick access to information and efficient use of resources. In addition, the built-in nature of SQLite allows you to use the database without the need to install an additional server or complex configuration. Also, this database will allow you to perform queries on any device (that is, it supports cross-platform compatibility). This database is a free and open database, which makes it available to a wide range of developers at no additional cost. This is important in the context of economic efficiency and availability of technology [28, 29]. Therefore, a compact built-in SQLite database was chosen to record the statistics of the number of people in the mask. Having statistical data about the number of masked people stored in a SQLite database, there is a need to process this data, including detecting people without a mask and notifying the administrator via messenger [28].

The system for recognizing masks on a person's face is also equipped with additional functionality, in particular, notifying an administrator or security guard about an intruder (a person without a mask). Such a system is easy to integrate into control modules at enterprises or checkpoints in public places, and can also be used for object recognition on video or in the development of real software ecosystems [4, 33–37]. This feature can be implemented using TelegramBotApi, a built-in HTTP-based interface created specifically for developers from the popular Telegram messenger, which is designed for writing bots. To create your own bot, you can use the "BotFather" bot already created by Telegram developers. To do this, you need to give the bot name, description, short name for future search in Telegram and API token for future introduction into the system through ready-made buttons (according to Figure 5).



/newbot 18:08

Alright, a new bot. How are we going to call it? Please choose a name for your bot. 18:08

MaskDetector Bot 18:08

Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot. 18:08

myown_mask_detector_bot 18:09

Done! Congratulations on your new bot. You will find it at t.me/myown_mask_detector_bot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:
6907438091:AAG_6D_M_fWahmTqT1sVofuWY11FTceLd2U
Keep your token **secure** and **store it safely**, it can be used by anyone to control your bot.

For a description of the Bot API, see this page:
https://core.telegram.org/bots/api 18:09

Figure 5 Creating a bot in the Telegram messenger.

For integration into various modules at enterprises and checkpoints, it is necessary to have any device based on Android or iOS, where you can freely download the developed application and place it at the entrance to the enterprise. For example, any enterprise can place an Android-based video intercom on which the application will be launched. A person who comes to work needs to look into the video intercom camera and receive a signal that the mask on the face has been found, and then the checkpoint is activated and allows the employee to enter the enterprise. A person without a mask will not be able to enter the enterprise, and the system will notify the person responsible for security in a Telegram bot about the presence of a person without a mask. Also, this neural network model is trained to recognize faces at an angle, so such a system can be integrated into public places and record the number of people with and without a mask for statistics.

The purpose of creating a Telegram bot is to make it easy to set up such a system and design a frontend for the application that will allow users to easily use the interface and menu items consisting of buttons. This action is automated during the deployment of the system, which will allow the user to receive a link to the bot that can be activated by the user to start working with it immediately when the application is installed on the device.

After creating the bot, this API-token is written to the system configuration file and used further to control it. The Telegram Bot API allows developers to quickly develop the backend and frontend (in the form of a messaging bot) and ensure uninterrupted functionality, as Telegram servers always work without severe interruptions and the bot server can be hosted directly on the device, which prevents unnecessary hosting costs. The result of the developed bot is shown in Figure 6.

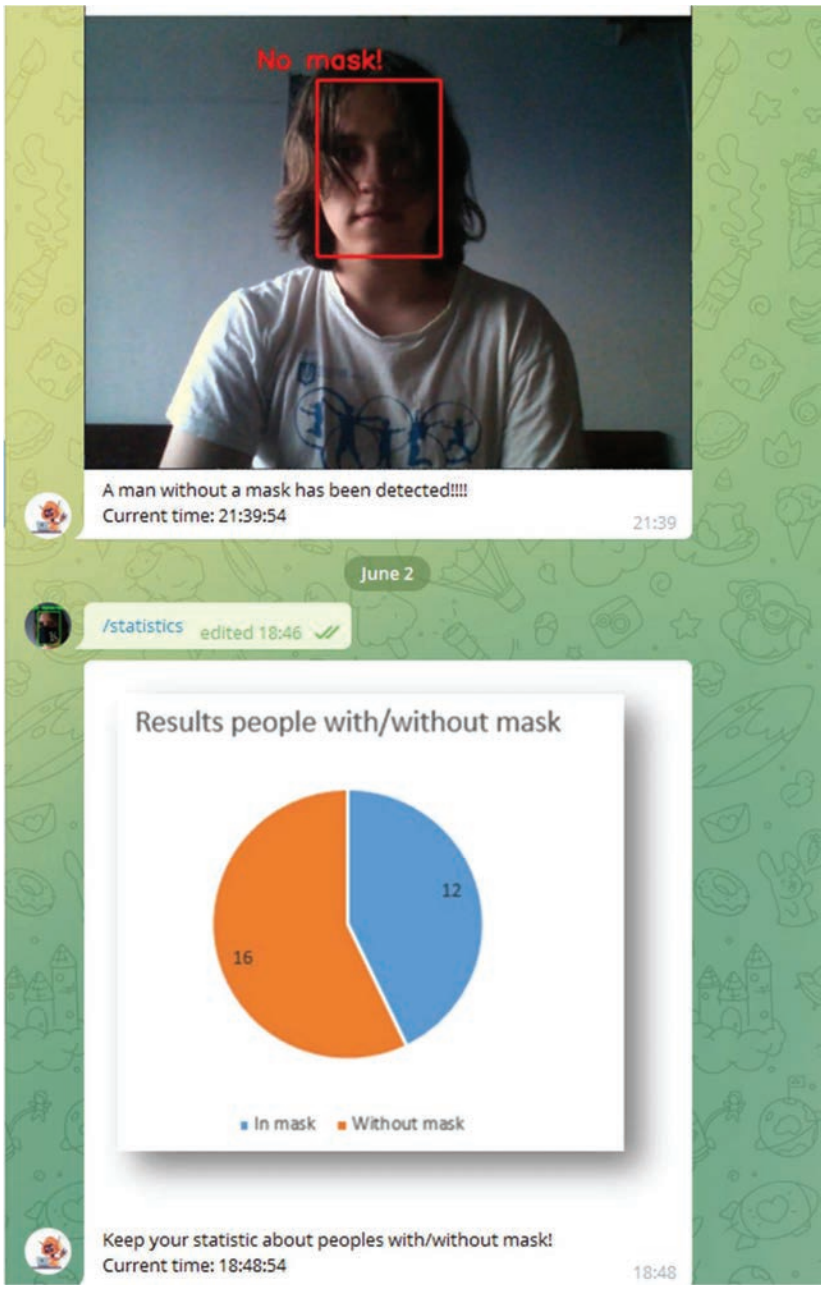
The bot notifies the administrator or security guard in a telegram (as indicated by a certain user_id in the system settings). Also, through this bot, if necessary, the user can find out general statistics at any time by simply entering the necessary command.

The next function is a sound notification. This function is necessary to provide a means of notifying a person about a pass or a ban on access to a public institution or the premises of an enterprise with dangerous substances. The notification function can be implemented using available sound processing libraries (PYO, PyAudio, Dejavu, PyDub, speech-dispatcher) [33, 38–40]. To call a sound notification, you need to create two objects of the winsound class. These objects accept the following parameters: (a) "duration" is the duration of the sound, "Alert" is the sound of a person warning about the absence of a mask, "Confirmation" is the sound of permission to enter the room. These beeps will be played once to warn of the absence of a mask or permission to pass.

There are two ways to deploy a project on a user's end machine: (1) run the ".exe" file format for easy launch, (2) download the code from GitHub and follow the installation steps. After starting, the user sees a menu (according to Figure 7) where you can specify parameters and start the camera.

The file with the ".exe" extension will allow the user to deploy the created application on the Windows operating system without additional actions, just run it and the necessary components will be downloaded and the user will be able to use this application. On other operating systems, you need to use the GitHub image, where the user will need to follow the steps to deploy the application on their device. Separate ".apk" and ".IPA" files have been developed for Android and iOS applications.

In Figure 7, the following elements can be distinguished: (1) the user ID in the Telegram messenger, where notifications about a person without a mask



A man without a mask has been detected!!!!
Current time: 21:39:54

Keep your statistic about peoples with/without mask!
Current time: 18:48:54

Figure 6 The result of the developed bot.

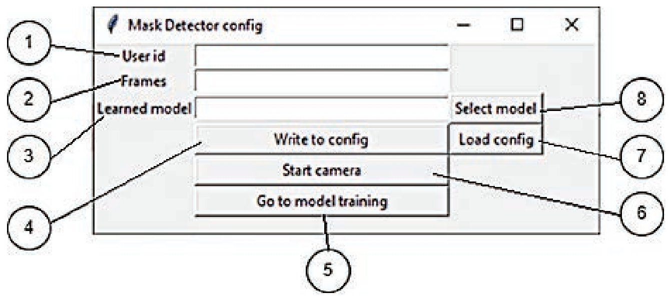
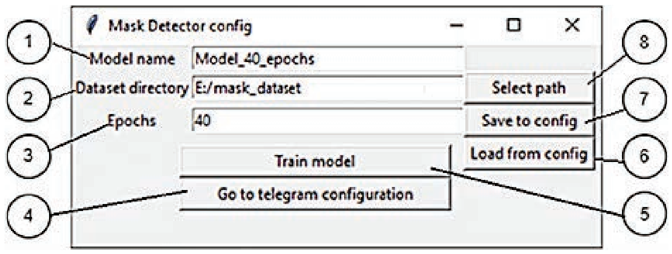


Figure 7 The start window for setting the parameters of the developed system.



Model name: Model_40_epochs
Dataset directory: E:/mask_dataset
Epochs: 40

Figure 8 The second window for setting the parameters of the developed system.

will be sent when the system is working, (2) the number of frames per second during the operation of the camera to recognize the presence of a mask on the face (if not set, then it will be 60), (3) a trained model that will be used by the NN to recognize the mask on the face, (4) a button to save the above data that can be changed, (5) go to the second settings window, (6) start the camera for recognition, (7) load the last saved data from the configuration file, (8) select the trained NN model.

The second window includes control elements for learning NN with the setting of the necessary parameters (according to Figure 8) [1, 5, 41, 42].

Figure 8 shows the following elements of the system: (1) the name of the NN model, (2) the path to the dataset on which the NN will be trained, (3) the number of epochs to train, (4) return to the start window of the system, (5) start model training, (6) load data from the configuration file, (7) save the entered data in the configuration file, (8) choose the path through the explorer to the dataset.

The process of facial mask recognition will be shown using the ready-made parameters from the configuration file (a model trained in 40 epochs). An attempt to bypass the protection of the mask recognition system on a

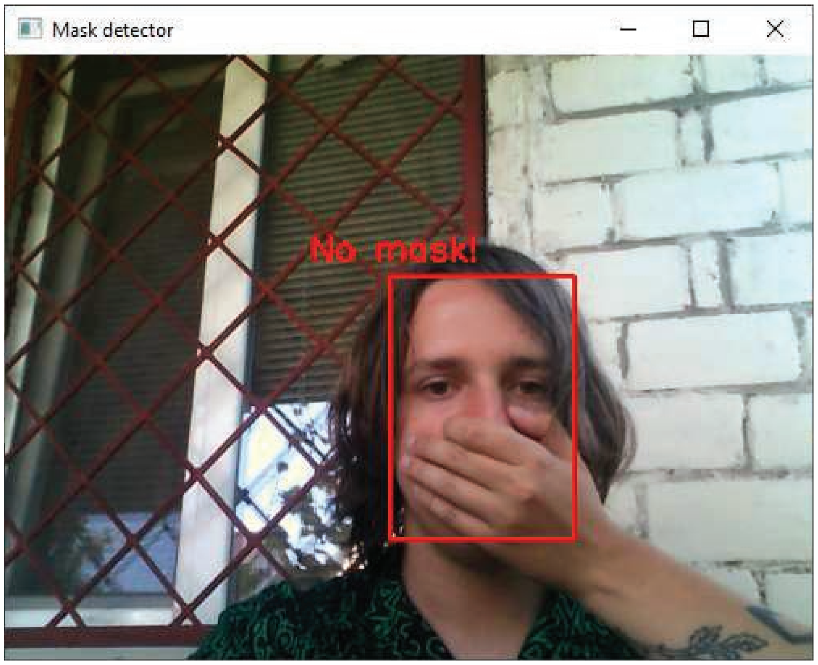


Figure 9 An attempt to bypass the protection of the developed system.

person's face, thereby covering the nose and mouth with a hand, will be recognized as "No mask!" (according to Figure 9).

The result of recognizing the absence of a mask on the face (according to Figure 10).

The result of recognizing the presence of a mask on a person's face (according to Figure 11).

Recognition of a mask of a different color with decorative elements (according to Figure 12).

The desktop application has many advantages when used in complex desktop systems when it is necessary to reach a large number of people. Otherwise, it is advisable to develop and use a mobile version of the application, this will facilitate the integration of the system in variable locations. The mobile app is a separate system that can work independently of the desktop app, the main thing is to register the bot and open it on any device that supports Telegram. The Windows app can also use a connected webcam and provide the same functionality as the mobile app.

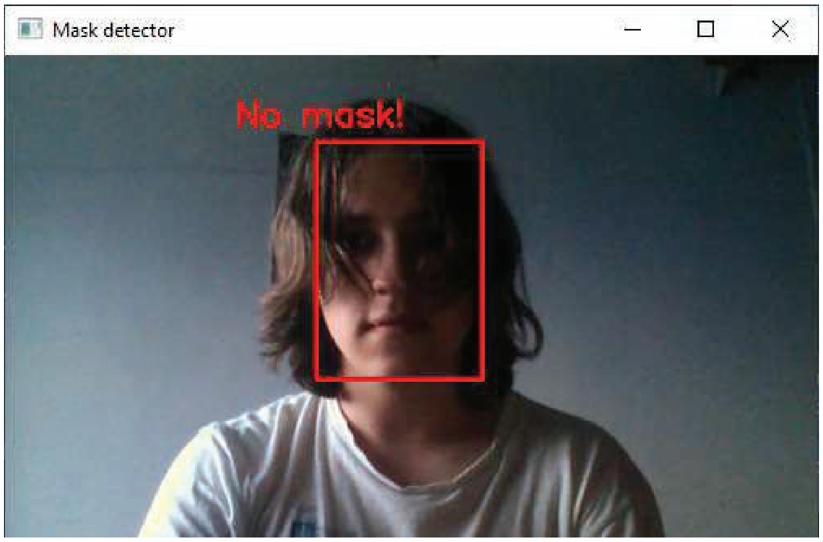


Figure 10 The absence of a mask on the face.

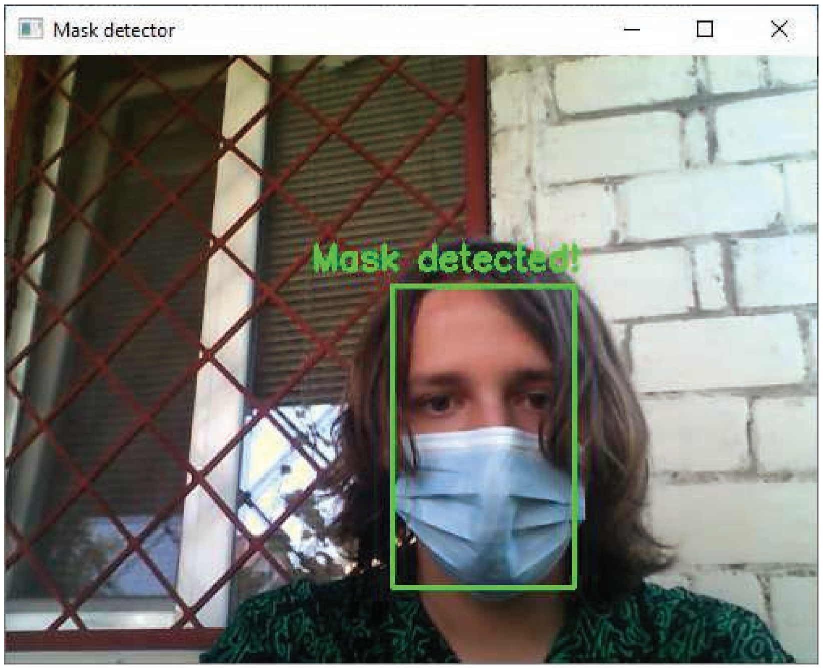


Figure 11 The presence of a mask on a person's face.

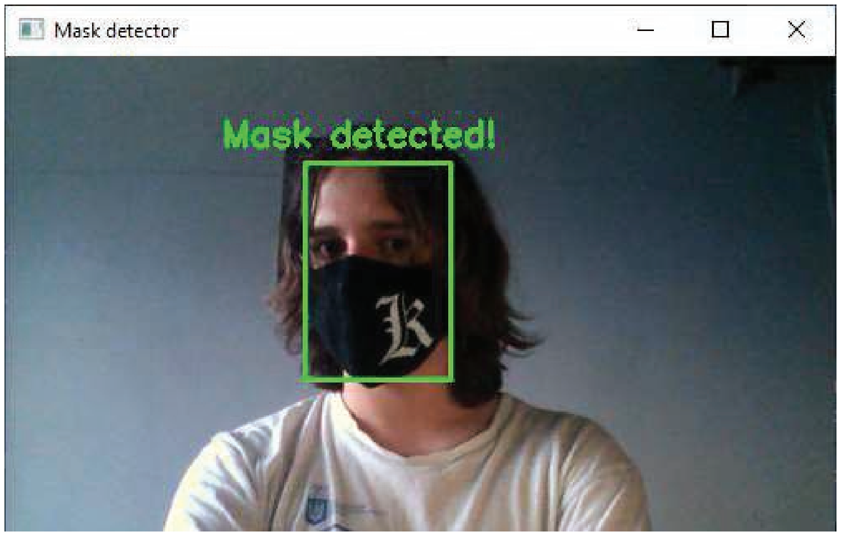


Figure 12 Recognition of a mask of a different color with decorative elements.

The authors of the work also developed a mobile version of the system for recognizing a mask on a person's face [43–47]. Figure 13a shows the main launch window of the "Mask Detection" application, decorated with a logo with the name of the application and a "Start" button that launches the program. This button allows the user to start using the face mask recognition application and takes the user to the main menu (according to Figure 13b). Figure 13b shows the main menu of the application, which has four buttons, they determine the functionality of the application: (a) the button "Camera Mode" allows the user to use the camera of his smartphone in real time to recognize the mask on the face, (b) the button "Upload Photo" allows you to upload your own photo for further recognition of the mask on it, (c) the "Neural Network" button provides an opportunity to select a neural network model for the application, (d) the "Settings" button allows users to adjust the parameters of the application according to their needs. When the user clicks on the "Camera Mode" button (according to Figure 13b), he sees his face on the smartphone screen, which will be highlighted by a green rectangle with the inscription "Mask Detected!", if he is wearing a mask (according to Figure 13c). Below the image is an information, which states that the CNN model is used for recognition, which recognized the mask on the face with 98.4% accuracy, which was trained over 200 epochs.

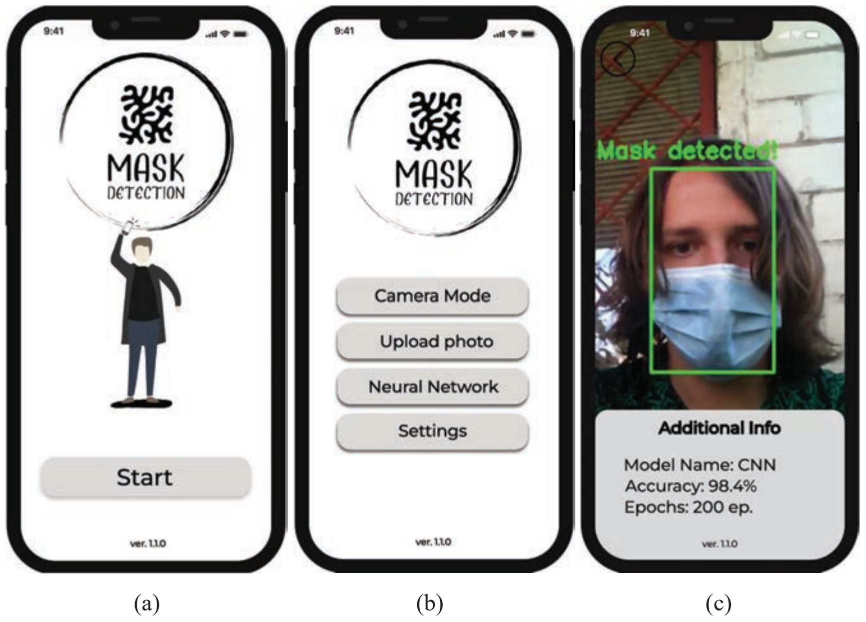
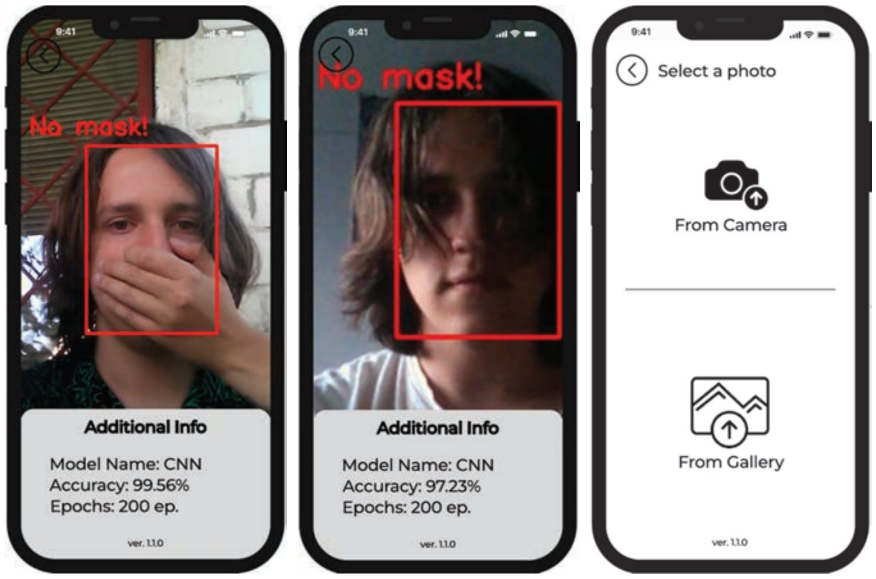


Figure 13 Windows of the developed mobile application: (a) start window, (b) window with main menu, (c) window of face mask recognition in the “Camera Mode”.

Figures 14a and 14b show the situation when the user is without a mask and the system highlights his face with a red frame and displays the inscription “No Mask!”. In addition, Figure 14a shows a situation when the system recognizes an attempt to bypass its protection by covering the face with a hand. Figure 14c shows the “Upload Photo” function from the main menu (according to Figure 13b), which allows the user to choose one of two options. He can choose to upload a photo from his smartphone gallery or take a photo in real time using the camera. After selecting one of these options, the user goes to the window (Figure 15a), where the mask recognition process on the selected photo takes place and all information about the selected neural network and its recognition accuracy on the given photo is displayed. The trained neural network model also makes it possible to recognize multiple faces in the input image, so the user can upload a collective photo with more than one person in it. When clicking the “Neural Network” button from the main menu (according to Figure 13b), the user goes to a window where he can select a specific neural network model (according to Figure 15b). The first model is “Convolutional Neural Network”, which is trained on 200 epochs. This model allows you to recognize a person’s face



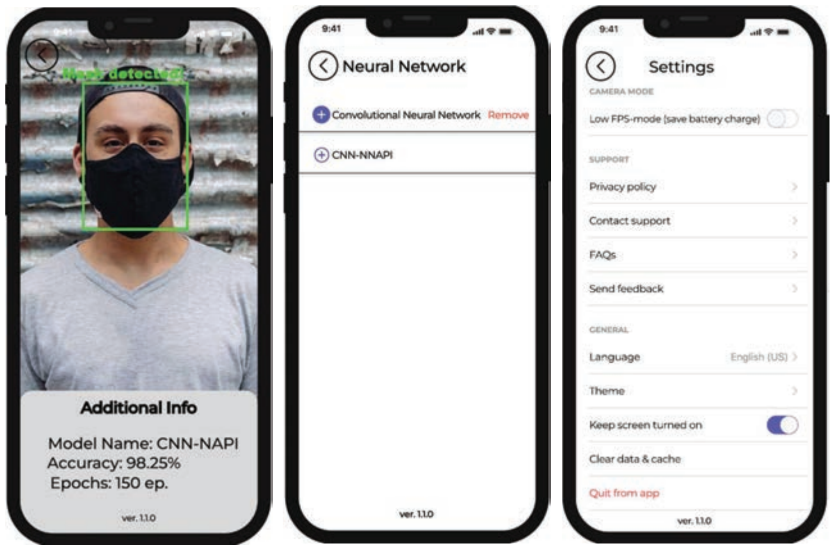
Model Name: CNN
Accuracy: 99.56%
Epochs: 200 ep.

Model Name: CNN
Accuracy: 97.23%
Epochs: 200 ep.

Figure 14 Windows of the developed mobile application: (a) window when the user is without a mask, (b) window with additional info, (c) window of the "Upload Photo".

with high accuracy and detect whether he is wearing a mask, but this neural network model is very difficult and not all smartphones have the ability to work well in real time without losing frames. Therefore, a lighter version of the model called "CNN-NNAPI" was created to replace it, which was also imported into the application. This network works with slightly less accuracy, but allows you to transmit a video stream of 60 frames without loss. After choosing a model, the user goes to the main menu, where he can then use the application and use this model for recognition.

When the user goes to the "Settings" menu from the main menu (according to Figure 13b), he has the opportunity to customize the application to his needs. in particular (according to Figure 15c): (a) choose "Low FPS Mode", which allows you to save battery power while using the application by limiting it to 30 frames per second, which significantly saves smartphone resources, (b) read the privacy policy, (c) leave feedback about the application by selecting "Send feedback" or report a critical bug, (d) choose the language of the application (among the available English and Ukrainian), (e) select the "Keep screen turned on" mode so that the smartphone screen does not turn off when the application is running in real time (can be turned off to save battery



Additional Info

Model Name: CNN-NAPI
Accuracy: 98.25%
Epochs: 150 ep.

(a) (b) (c)

Figure 15 Windows of the developed mobile application: (a) window with face mask recognition based on the uploaded photo, (b) window with neural network model selection, (c) window with application settings.

power), (f) clear the data and cache of the application to remove all uploaded photos and other information; (g) press the exit button from the application.

Testing of the developed mobile application was conducted among many experts in this field. it was repeatedly noted that the application works quickly and clearly recognizes a mask on a person's face, while it can collect statistics and report when a person enters the premises without a mask. It is worth noting that the mobile application is still being tested in real conditions and is being refined according to experts' comments.

The use of a mobile application expands the possibility of using this system to recognize a mask on a person's face. The developed mobile application for recognizing a mask on a human face is an easy-to-use, informative, powerful tool using machine learning technologies, but at the same time it does not overload the operating system of the mobile device. The mobile application can be easily integrated with other mobile devices, including mobile cameras, mobile robots, etc.

In the future, it is planned to add to the system the possibility of transferring statistics and messages through other messengers. In addition, it is planned to teach a neural network to recognize an incorrectly worn mask.

5 Conclusions

The developed system has a fairly wide range of applications and impacts on public health and safety. It can be successfully used in the subway, industrial enterprises, government institutions, educational institutions, offices and other public places.

The developed system recognizes and records statistics about the presence of a mask on a person's face with the help of neural networks, notifies about a person without a mask in the Telegram messenger with an appropriate sound signal. Such capabilities of the system will make it possible to identify violators and ensure a high level of security in public places and reduce the level of morbidity among the population.

The key technologies used in the system are the MobileNetV2 neural network, libraries for developing a neural network and its training, including Tensorflow, Sklearn, Imutils, Virtualenv, Numpy, the Tkinter library for forming and creating a cross-platform GUI, the Bot API for Telegram, OpenCV for working with cameras in the system.

For testing, a separate independent dataset was collected that was not part of the dataset for training, validation, and testing. After that, the system was tested on the desktop and mobile versions and the accuracy of the system was determined, and the accuracy was also checked on the augmented data.

Based on the tasks and selected technologies, this system takes up little space on PCs, mobile devices or microcontrollers, has a clear interface and extensive functionality, and will be distributed with open-source code and free of charge. Overall, such a model offers numerous benefits, ranging from increased accessibility and transparency to collaborative innovation and cost savings.

References

[1] A. Sheremet, Y. Kondratenko, I. Sidenko, G. Kondratenko, 'Diagnosis of Lung Disease Based on Medical Images Using Artificial Neural Networks', 3rd Ukraine Conference on Electrical and Computer Engineering, Lviv, Ukraine, 2021.

[2] D, Chumachenko, et al., 'Forecasting of COVID-19 Epidemic Process in Ukraine and Neighboring Countries by Gradient Boosting Method', in: E, Faure, et al. (eds) Information Technology for Education, Science, and Technics. Lecture Notes on Data Engineering and Communications Technologies, 178, Springer, Cham, 2023.

[3] S. Thuseethan, et al., ‘Deep COVID-19 Recognition Using Chest X-ray Images: A Comparative Analysis’, International Conference on Artificial Intelligence, Colombo, Sri Lanka, 2021.

[4] A. Basu, M. F. Ali, ‘COVID-19 Face Mask Recognition with Advanced Face Cut Algorithm for Human Safety Measures’, 12th International Conference on Computing Communication and Networking Technologies, Kharagpur, India, 2021.

[5] J. Luo, R. Luo, ‘Research on Image Recognition based on Reinforcement Learning’, 4th International Conference on Computer Vision, Image and Deep Learning, Zhuhai, China, 2023.

[6] Y. Xing, W. Cai, ‘Mask recognition system based on anchor’, 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference, Chongqing, China, 2022.

[7] G. Deore, R. Bodhula, V. Udpikar, V. More, ‘Study of masked face detection approach in video analytics’, Conference on Advances in Signal Processing, Pune, India, 2016.

[8] P. P D, P. Nath Singh, ‘Masked & Unmasked Face Recognition Using Support Vector Machine Classifier’, International Conference on Mobile Networks and Wireless Communications, Tumkur, Karnataka, India, 2021.

[9] B. Kocacinar, et al., ‘A Real-Time CNN-Based Lightweight Mobile Masked Face Recognition System’, in: IEEE Access, 10, 2022.

[10] L. Cimmino, et al., ‘M2FRED: Mobile Masked Face REcognition Through Periocular Dynamics Analysis’, in: IEEE Access, 10, 2022.

[11] NQ. Dao, et al., ‘Management of Video Surveillance for Smart Cities’, in: Handbook of Smart Cities, Springer, Cham, 2018.

[12] M. Pudyel, M. Atay, ‘An Exploratory Study of Masked Face Recognition with Machine Learning Algorithms’, SoutheastCon 2023, Orlando, FL, USA, 2023.

[13] S. Hao, C. Chen, Z. Chen, K.-Y. K. Wong, ‘A Unified Framework for Masked and Mask-Free Face Recognition Via Feature Rectification’, International Conference on Image Processing, Bordeaux, France, 2022.

[14] W. Lin, et al., ‘Masked Face Recognition with Qaudruplet Loss’, International Conference on Image Processing and Computer Applications, Changchun, China, 2023.

[15] M. Zhang, R. Liu, D. Deguchi, H. Murase, ‘Masked Face Recognition With Mask Transfer and Self-Attention Under the COVID-19 Pandemic’, in: IEEE Access, 10, 2022.

[16] M. Mobaraki, et al., 'Masked Face Recognition Using Convolutional Neural Networks and Similarity Analysis', 24th International Conference on Digital Signal Processing, Rhodes, Greece, 2023.

[17] Y. Kondratenko, et al., 'Machine Learning Techniques for Increasing Efficiency of the Robot's Sensor and Control Information Processing', in: Sensors, 22(3), 2022.

[18] H. Wang, et al., 'Research on Smooth Edge Feature Recognition Method for Aerial Image Segmentation', 15th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics, Beijing, China, 2022.

[19] S. Zhang, 'Character Recognition of Historical and Cultural Relics Based on Digital Image Processing', 5th International Conference on Electronics, Communication and Aerospace Technology, Coimbatore, India, 2021.

[20] W. Yu, et al., 'Application Research of Image Feature Recognition Algorithm in Visual Image Recognition', Conference on Telecommunications, Optics and Computer Science, Dalian, China, 2022.

[21] T. Biloborodova, et al., 'ECG Classification Using Combination of Linear and Non-Linear Features with Neural Network', in: Studies in Health Technology and Informaticsthis link is disabled, 2022.

[22] B. Brosnan, I. Skarga-Bandurova, T. Biloborodova, I. Skarha-Bandurov, 'An Integrated Approach to Automated Diagnosis of Cervical Intraepithelial Neoplasia in Digital Histology Images', Studies in Health Technology and Informaticsthis link is disabled, 2023.

[23] V. Alekseeva, et al., 'Intelligent Decision Support System for Differential Diagnosis of Chronic Odontogenic Rhinosinusitis Based on U-Net Segmentation', in: Electronics, 12(5), 2023.

[24] M. Tetiana, Y. Kondratenko, I. Sidenko, G. Kondratenko, 'Computer vision mobile system for education using augmented reality technology', in: Journal of Mobile Multimedia, 17(4), 2021.

[25] O. Striuk, et al., 'Implementation of Generative Adversarial Networks in Mobile Applications for Image Data Enhancement', in: Journal of Mobile Multimedia, 19(3), 2023.

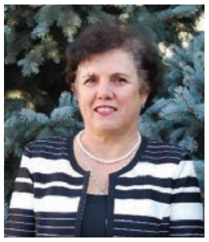
[26] C. Aggarwal, Neural Networks and Deep Learning, Springer, Cham, 2023.

[27] T. Adar, E. K. Delice, O. Delice, 'Detection of COVID-19 From A New Dataset Using MobileNetV2 and ResNet101V2 Architectures', Medical Technologies Congress, Antalya, Turkey, 2022.

[28] U. Kulkarni, et al., 'Facial Key points Detection using MobileNetV2 Architecture', 8th International Conference for Convergence in Technology, Lonavla, India, 2023.
[29] A. B. Handoko, et al., 'Evaluation of YOLO-X and MobileNetV2 as Face Mask Detection Algorithms', Industrial Electronics and Applications Conference, Kuala Lumpur, Malaysia, 2022.
[30] O. Striuk, Y. Kondratenko, I. Sidenko, A. Vorobyova, 'Generative Adversarial Neural Network for Creating Photorealistic Images', IEEE 2nd International Conference on Advanced Trends in Information Theory, Kyiv, Ukraine, 2020.
[31] I. Sova, I. Sidenko, Y. Kondratenko, 'Machine Learning Technology for Neoplasm Segmentation on Brain MRI Scans', CEUR Workshop Proceedings, PhD Symposium at ICT in Education, Research, and Industrial Applications, 2791, Kharkiv, Ukraine, 2020.
[32] I. Khortiuk, G. Kondratenko, I. Sidenko, Y. Kondratenko, 'Scoring System Based on Neural Networks for Identification of Factors in Image Perception', CEUR Workshop Proceedings, 4th International Conference on Computational Linguistics and Intelligent Systems, 2604, Lviv, Ukraine, 2020.
[33] N. Lidströmer, H. Ashrafian (Eds), 'Artificial Intelligence in Medicine', Springer, Cham, 2022.
[34] V.M. Kuntsevich, et al. (Eds), 'Control Systems: Theory and Applications', River Publishers, Gistrup, Delft, 2018.
[35] R. Duro, et al. (Eds), 'Advances in intelligent robotics and collaborative automation', River Publishers, Aalborg, 2015.
[36] V. Lytvyn, et al., 'An intelligent system of the content relevance at the example of films according to user needs', International Workshop on Information-Communication Technologies and Embedded Systems, ICT and ES, 2516, 2019.
[37] S. Kryvyi, O. Grinenko, V. Opanasenko, 'Logical Approach to the Research of Properties of Software Engineering Ecosystem,' 11th International Conference on Dependable Systems, Services and Technologies, Kyiv, Ukraine, 2020.
[38] S. Putatunda, 'Practical Machine Learning for Streaming Data with Python', Apress, Berkeley, CA, 2021.
[39] N. Sanghi, 'Deep Reinforcement Learning with Python', Apress, Berkeley, CA, 2021.
[40] J. Unpingco, 'Python Programming for Data Analysis', Springer, Cham, 2022.

[41] A. I. Shevchenko, 'Natural Human Intelligence – the Object of Research for Artificial Intelligence Creation,' IEEE 14th International Conference on Computer Sciences and Information Technologies (CSIT), Lviv, Ukraine, 2019.
[42] A. I. Shevchenko, M. S. Klymenko, 'Developing a Model of "Artificial Conscience",' IEEE 15th International Conference on Computer Sciences and Information Technologies (CSIT), Zbarazh, Ukraine, 2020.
[43] T. Green, J. Labrecque, 'A Guide to UX Design and Development', Apress, Berkeley, CA, 2023.
[44] M.E. Auer, T. Tsiatsos (Eds), 'New Realities, Mobile Systems and Applications', Springer, Cham, 2022.
[45] M. Dakić, 'Mobile App Development for Businesses', Apress, Berkeley, CA, 2023.
[46] J. Singh, D. Das, L. Kumar, A. Krishna (Eds), 'Mobile Application Development: Practice and Experience', Springer, Singapore, 2023.
[47] A. Alnoor, K.K. Wah, A. Hassan (Eds), 'Artificial Neural Networks and Structural Equation Modeling', Springer, Singapore, 2022.

Biographies



Galyna Kondratenko is an Associate Professor, Ph.D., Associate Professor of the Intelligent Information Systems Department, Senior Researcher at Petro Mohyla Black Sea National University, Ukraine. She is a specialist in control systems, decision-making, fuzzy logic. She worked in the framework of international scientific university cooperation during the implementation of international projects with the European Union: TEMPUS (Cabriolet), Erasmus + (Aliot) and DAAD-Ostpartnerschaftsprogramm (project with the University of Saarland, Germany). Her research interests include computer control systems, fuzzy logic, decision-making, intelligent robotic devices.

Ievgen Sidenko is an Associate Professor, Ph.D., Associate Professor of the Intelligent Information Systems Department at Petro Mohyla Black Sea National University (PMBSNU), Ukraine. He has received master degree in speciality "Intelligent decision making systems" (2010) at PMBSNU and Ph.D. degree in "Information technologies" (2015) at PMBSNU. His research interests include fuzzy sets and fuzzy logic, decision-making, optimization methods, neural networks, data mining, clustering and classification.

Saliutin Maksym is a master's student of the Intelligent Information Systems Department at Petro Mohyla Black Sea National University (PMBSNU). He received his bachelor's degree with a major in 122 "Computer Science" at PMBSNU. He is engaged in creation of automation systems, as well as in development of neural network models. He is interested in studying new technologies in the field of machine learning and neural networks.

Yuriy Kondratenko is Doctor of Science, Professor, Honour Inventor of Ukraine (2008), Corr. Academician of Royal Academy of Doctors (Barcelona, Spain), Head of the Intelligent Information Systems Department at Petro Mohyla Black Sea National University (PMBSNU), Leading Researcher at the Institute of Artificial Intelligence Problems under MES and NAS of Ukraine. He has received (a) the Ph.D. (1983) and Dr.Sc. (1994) in Elements and Devices of Computer and Control Systems from Odessa National Polytechnic University, (b) several international grants and scholarships for conducting research at Institute of Automation of Chongqing University, P.R.China (1988–1989), Ruhr-University Bochum, Germany (2000, 2010), Nazareth College and Cleveland State University, USA (2003), (c) Fulbright Scholarship for researching in USA (2015/2016) at the Dept. of Electrical Engineering and Computer Science in Cleveland State University. Research interests include robotics, automation, sensors and control systems, intelligent decision support systems, fuzzy logic.