2020 24th International Conference on System Theory, Control and Computing (ICSTCC)

Implementing an embedded system to identify possible COVID-19 suspects using thermovision cameras

Adrian Florea
Department of Computer Science and Electrical Engineering
Lucian Blaga University of Sibiu
Sibiu, Romania
ORCID 0000-0003-0278-4825

Valentin Fleaca
Department of Computer Science and Electrical Engineering
Lucian Blaga University of Sibiu
Sibiu, Romania
valentinfleaca@gmail.com

***Abstract*—The main goal of this paper is to prove that by combining thermal vision cameras and image processing with many deep learning classification algorithms we developed an effective embedded system with high applicability in this critical period caused by COVID-19 pandemic disease. Using fixed and mobile thermal cameras we envisioned and developed a real time temperature screening capable of sending alarm signals over network or by SMS to local authorities along with multiple detection metrics such as the age, the gender, the facial emotion, the GPS location where the alarm went off, the temperature reading from the human face and also if the subject is wearing or not a medical face mask.**

Keywords— thermal, camera, application, detection, mask

I. INTRODUCTION

A novel infection with the coronavirus was first found in the city of Wuhan, in the province of Hubei in China. The world Health Organization (WHO) has officially announced the identification of a new virus which will be called by 2019-nCoV in January 2020 [1] and has recognized that the virus can cause respiratory disease with cough, fever and pneumonia. The WHO Emergency Committee declared on 30 January the novel coronavirus infection as a pandemic disease due to the rapid human-to-human transmission.

Anyone who is infected with coronavirus may have typical symptoms such as dry cough, fever, fatigue and in some cases nasal congestion, pain, sore throat or diarrhea. The infection could be fatal for older people or for someone with a chronic condition. As of now (May 2020), no effective vaccine for COVID 19 was developed. As the recorded death and infected people cases continue to increase [2], lockdowns have been conducted by several nations to reduce the spread of coronavirus impact. They often attempt to identify a potential infected person from crowds by using an infrared thermometer to track temperatures in public places. But there are drawbacks to the use of such an infrared thermometer gun, as it may not cover all the people, it can also favorize the spreading of the virus, as the user of the infrared thermometer gun has to do a close range (5 to 15 cm) one by one person at a time, which is time consuming and could lead to a high number of people queuing. To prevent these drawbacks, an alternative technology is needed.

As most of the countries are lifting some of the restrictions, some governments are enforcing wearing of a medical mask when going out in the community, especially when using the public services of transportation or when entering shops, schools, offices of administrative and municipalities, workplaces, usually indoor places. Thus, efficient real time face mask detection is needed.

This work started in the context of workplace safety and employees emotion recognition. We developed and integrated a Face Emotion Recognition (FER) system with a new module that detects the temperature of people. The implemented embedded system can identify possible COVID-19 suspects using thermal vision (infrared) cameras. It is capable, besides the human body temperature detection, to also detect the age, gender, emotion and if the person is wearing a mask or not, using a smartphone. Different characteristics for the thermal vision cameras have been tested, such as: emissivity and distance between the subject and the camera in order to calibrate the camera to reflect the human body temperature as accurate as possible.

First, we designed, implemented and evaluated a new system of recognition of facial expressions using different machine learning techniques that would ultimately perform better in terms of computational speed and accuracy than existing methods. The FER system accepts as input a live video stream and it is able to detect the human face and classify the facial expression into one of the seven basic human face emotions (neutral, happy, sad, surprised, disgusted, angry, and afraid). In our opinion losing control of emotions means losing control of safety. For example, if a human machine operator supervises or controls multiple heavy duty machines then it is quite important for that human to not be in emotional distress (angered, shocked or tired). The developed FER system provides notifications sending alerts when a human should not perform a dangerous task under emotional distress. Also, such an application could verify that a construction, civil engineering, industry worker is equipped accordingly with protection gear. Or, if a worker that supervises an industrial assembly line or maneuvers an excavator does not feel well (looks distressed/ angry/sick), it could create multiple damages if not stopped or assisted before or during the task he is performing.

Second, the module that reads the human body temperature was developed in two scenarios: first, using a small camera attached to an Android smartphone for mobile use and second, using a fixed camera as a desktop application with remote controlling possibility from longer distances being able to simultaneously read temperatures from multiple subjects.

The rest of the paper is organized as follows: section 2 describes the methodology applied in the development of the two software applications (mobile and desktop) highlighting their features and applications workflow. Section 3 briefly presents the detection and classification algorithms and the datasets used for training. The fourth section of the paper provides hardware and software system requirements. Some limitations and implications for practice of the system are

described in section 5. The final section presents general conclusions, together with the main contributions and further work.

II. METHODOLOGY

This section describes the work flows of two different but with similar purpose systems. The first one is a proposed handheld solution for effective temperature screening using a thermal vision camera attached to a personal smartphone. The second one involves a fixed thermal camera controlled via an internet connection with broadcasting possibility to multiple clients. This proposed embedded system was implemented and tested in the campus of the Engineering Faculty, Lucian Blaga University of Sibiu, Romania.

A. EmoTemp - Android application

Using a thermal camera for smartphones (Flir One Pro) we envisioned and developed a real time temperature screening capable of sending alarm signals over network or by SMS to local authorities along with multiple detection metrics such as the age, the gender, the facial emotion (one of the seven basic human emotions: sadness, happiness, anger, disgust, surprise, afraid, neutral), the GPS location where the alarm went off, the temperature reading from the human face and also if the subject was wearing or not a medical face mask.

The application analyzes and displays frames in real time at roughly five frames per second, which is enough to capture basic human movements such as walking or even jogging. The user of the application has the possibility to alter the emissivity setting for the thermal camera with a rational number between zero and one. The emissivity of a material surface represents its effectiveness in emitting energy as thermal radiation. The human skin emissivity ranges from 0.95 up to 0.999 [3]. During tests performed, while also using regular body thermometers, we found out that the emissivity value has to be set around 0.98 for best results, as this value will give the closest temperature readings to the regular thermometer value reading of the human body. We also studied how the distance between the camera and subject influences the temperature readings. Starting from a distance of more than 7 meters, a human body emitted, with the emissivity configuration set to 0.98, a maximum temperature of 34.59 degrees Celsius to a distance of about 30 centimeters a maximum temperature reading of 37.07 degrees Celsius. The test showed how the distance affects the temperature reading, having a 2.5 Celsius degrees temperature difference between 7 meters and 30 centimeters. However, the temperature sensors of cameras are not sensitive at less than two meters providing the same temperature both at 20cm and 200cm. Thus, we concluded that the temperature can be read correctly even at a distance of 2 to 2.5 meters, increasing this way the field of view (FoV) of the camera so that multiple subjects can be screened at once.

We also validated the application against real life scenarios such as screening the temperature of football players before entering the pitch and also for restaurant waiters to read the temperatures and emotions of customers before and during their stay. The application proved to be valuable for the waiters because they no longer need to get too close to a potentially infected customer.

Fig. 1. Example of automatic finding of the points with the highest temperature. Warning when a value is above the set threshold temperature (here it was set at 37.0 degrees Celsius). The temperature point from the nose turned red.

Fig. 2. Screenshot during real time application running in dual mode (thermal image in the background, visible image the foreground).

Figure 2 illustrates how the thermal vision camera captures infrared images which will be transformed into thermal images, while on the visible spectrum images the detections will be performed: age-white text („Adult" representing the category [25-32] years old); gender-blue circle („Male"; red circle means „Female"); emotion-red text („Happy"); mask detection-black text („Mask" + detection confidence [0-1]. The visible light image will be displayed in the foreground while maintaining the thermal image in the background if the option to display the processed image is also checked. The visible light image will be taken from the device's camera. This implies the need of mapping of the temperature point location from the thermal image to the visible light image. The application workflow is illustrated in figure 3.

When the application is started, it first checks the camera micro USB connection. Once the thermal vision camera is connected, the application will start capturing infrared images which will be transformed into thermal images having temperatures and different colors ready to be displayed on the device's screen. To increase the processing performance and also keeping in mind that the value read from the infrared sensor changes relatively slow, once every 3 to 4 frames, we decided that only every odd captured frame to find the maximum five highest temperatures from the frame. In all the other frames, this search will not be performed; instead the cached values will be used. Again, for the same processing performance reasons and also for the fact that the human face might have the lowest size of 35x35 pixels depending on the distance to the camera, we decided not to process every single pixel from the image, this would have meant that 640x480 pixels needed to be checked, but instead to process just specific pixels that fit on a predefined grid. This grid consists of 64x48 pixels evenly distributed from one to another on the image, resulting in fewer pixel temperature checks, every 64th pixel from every line and every 48th pixel from each column will be checked. If at least one of the five highest temperature values from the image is above the predefined threshold value, that can be set either at the start of the application or during runtime through a spinner, and also if the option to process images from visible light spectrum is checked, then the temperature point that is above the threshold will be mapped to the closest detected face and start the alarm signals.

Although the thermal vision camera has also a visible light camera module, it proved out to be a lot more efficient from the computational point of view to use an entire separate processing thread along with a separate, already existing visible light camera from the smartphone.

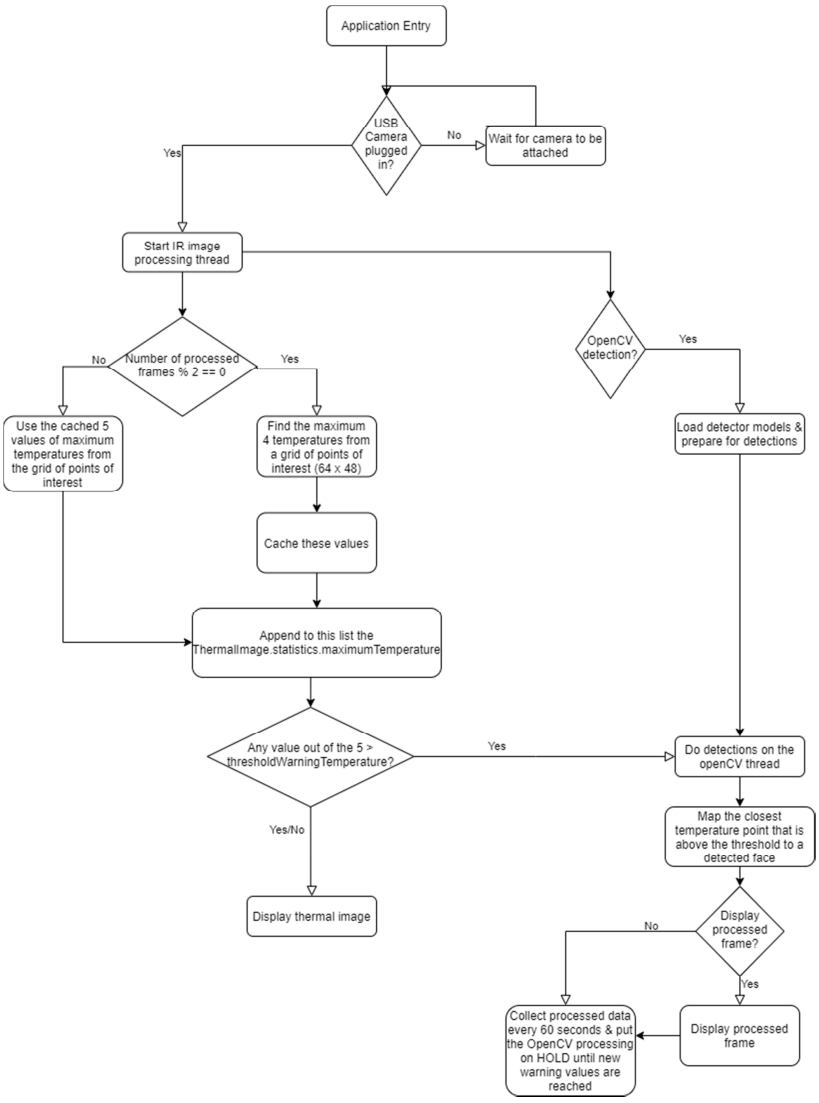

Fig. 3. Android application workflow

B. *Desktop application system*

This application has a similar functionality with the previous one but with some key differences. The most important difference being the fact that the thermal vision camera is now a fixed one (Flir A320) and that the detection of the presence or not of a medical mask being used on a human face was done directly over the thermal images live stream as this thermal camera model did not have a visible light camera module and attaching a visible light camera to the system would have increased the costs.

The architecture for this system consists of two modules. The first module represents a desktop application developed in .NET CORE 3.1, which handles the camera connection through Ethernet, it displays and processes the thermal images received from the camera and signals an alarm when a temperature reading from a predefined area is detected above a given threshold. The second module consists of a server and a client submodule. The server submodule was developed in .NET CORE 3.1 and has the role to stream the images processed by the first module over the local network, while the client submodule will receive this feed via web sockets through HTML5 and JavaScript's WebRTC library. The first module and the server submodule have to run on the same machine while the client sub module can be run on multiple different machines.

A sample illustration of the usage of the two modules to form a single system is illustrated in figure 4. The figure presents the synergy between the two modules, which for test purposes were running both on the same machine. Due to humidity, ambient temperature and emissivity settings at that point, the subject's maximum temperature was just that of 34.4 degrees Celsius. The threshold alert temperature was set to 34 degrees Celsius and this triggered an alert, starting the image processing thread. This thread processes the current thermal image and looks for human faces and if they are covered by a medical mask or not.

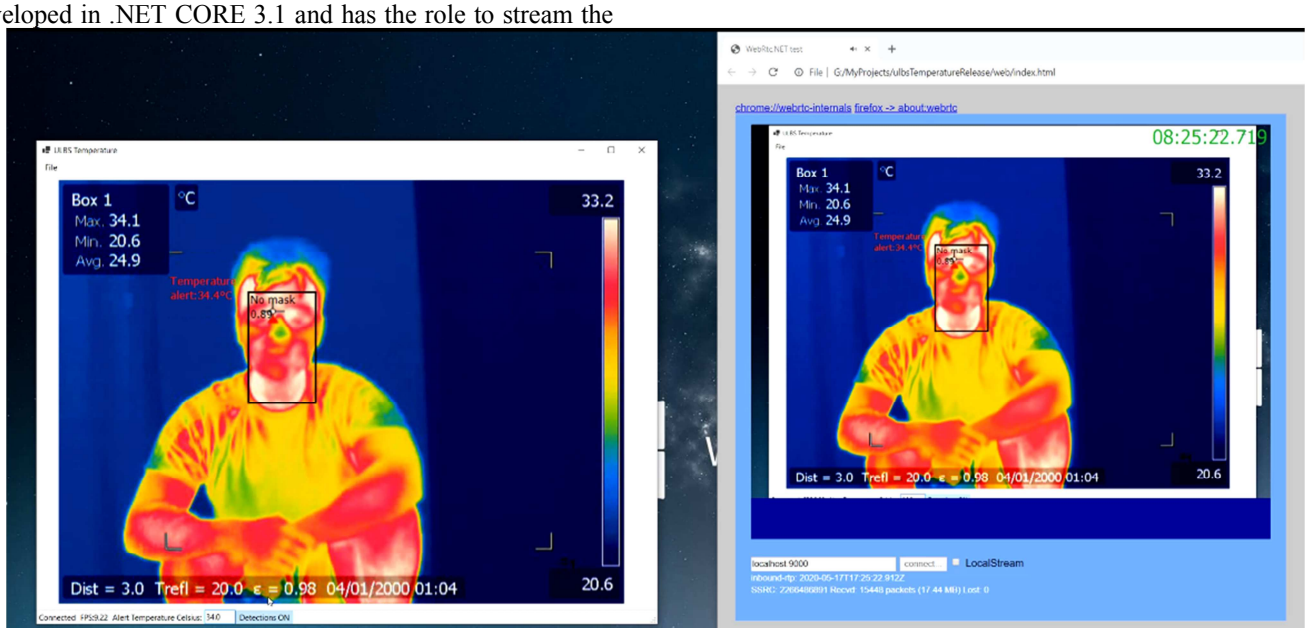

Fig. 4. Sample screenshot for the desktop application system. Left – the first module, having in background running the server submodule of the 2nd module. Right – the client submodule running in Google Chrome's web browser.

III. DETECTION ALGORITHMS

For the human face detection, the Viola-Jones [4] face detection algorithm was used. Deep learning methods like convolutional neural networks (CNNs) can deliver highly accurate classification results when feed with large enough data sets and respective labels. Thus, for the age and gender detection, the same convolutional neural network that has a similar architecture with AlexNet [5] was used. The first Convolutional Layer contains 96 filters of 7×7 pixels, the second Convolutional Layer contains 256 filters of 5×5 pixels, the third and final Convolutional Layer contains 384 filters of 3 × 3 pixels.

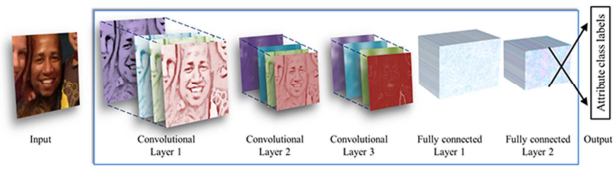

Fig. 5. CNN arhitecture for age and gender classification. Taken from [7].

Finally, two fully connected layers are added, each containing 512 neurons as taken from [7]. The network was pre-trained twice on the same Audience [6] dataset but each time with a different target output. First time gender detection for male or female, while the second time the age class, one age segment from [(0 – 2), (4 – 6), (8 – 12), (15 – 20), (25 – 32), (38 – 43), (48 – 53), (60 – 100)]). The dataset contains 26580 images. The structure for the CNN can be seen in figure 5.

The medical mask detection was performed using a CNN implemented by Google, MobileNetV2 [8] in the submodule of the framework TensorFlow, Keras. The dataset used for the training of the network is a synthetical one because there is not yet any dataset available at the global level. With the help of facial landmarks detection using an ensemble of regression trees [9], the human mouth was detected and on top of it a surgical mask was applied. Figure 6 illustrates the steps performed in order to generate such a synthetic dataset. Even though the model obtained close to 99% accuracy over the synthetic image dataset [10], we observed, in our tests, that it reaches above 90% accuracy for real time detection.

Fig. 6. Creation of the synthetical dataset. Step 1 – face detection; Step 2 - facial landmarks detection. Step 3 – Drawing on top of the original image a surgical mask. As taken from [10].

As for the face emotion recognition (FER) we used our own implementation of such a system. After a face is detected using the Viola-Jones face detection algorithm [4], we applied a facial landmarks predictor using an ensemble of regression trees [9] obtaining 68 2-dimensional points representing the facial landmarks positions on the face. From these points, we calculated 38 specific pair pixel Euclidian distances obtaining a vector of 38 1-dimension elements, which will be fed to a machine learning classifier algorithm. We tested with multiple algorithms belonging to the supervised learning cathegory and the ones giving the highest accuracies were a Multilayer Perceptron Classifier (MLPC) and close to it was a Support Vector Classifier (SVC). We trained the MLPC over two combined already face emotion labeled datasets, first being The Karolinska Directed Emotional Faces (KDEF) [11] having 490 images and the second one being The Extended Cohn-Kanade Dataset (CK+) [12] with 309 images. In order to find the best hyperparameters for both the MLPC and the SVC used an evolutionary computing algorithm for a hyperparameter search or tuning of a set of hyperparameters for the classifier algorithm.

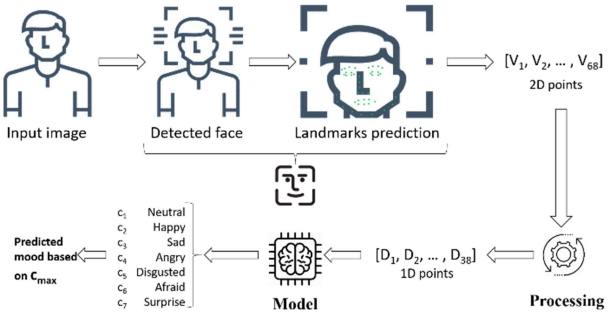

Fig. 7. Our own implementation of the FER system

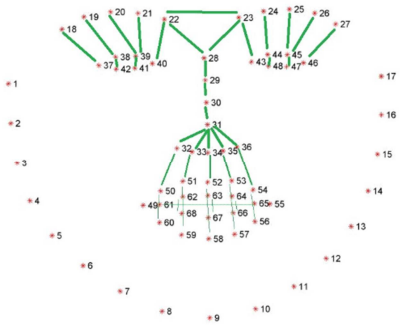

Fig. 8. Specific pair pixel Euclidian distances calculated by our FER marked with green lines between pixel indices retrieved by the landmarks predictor

For the MLPC, the hyperparameters included number of hidden layers, number of neurons on each layer, activation function for the hidden layer, the solver for weight optimization. While for the SVC the tunning was performed over kernel type and penalty parameter of the error term. The output of each classifier is a 7 elements 1-dimensional vector containing the confidences for each one of the seven basic human emotions. Figure 7 presents the entire workflow for our FER system, while figure 8 evidentiates the specific Euclidian distances calculated. The trained MLPC obtained 79% accuracy over the KDEF dataset when trained over 80% of the dataset and testing over the last 20%.

IV. System Requirements

The EmoTemp Android application was developed using Android Studio 3.5 integrated development environment (IDE) with Java SE 8 support. As for the libraries used, OpenCV4Android version 3.4.3 [13] and TensorFlowLite [14] are at the core. The application was deployed and tested successfully on a Samsung Galaxy TAB A T580 10.1-inch display having a 1920x1200 resolution, 2 GB RAM memory, octa core Exynos 7870 CPU at 1.6 MHz running on Android OS version 8.1.

The first module and the server submodule of the Desktop application system were developed in Visual Studio 2019 versions 16.6. The used programming language was C# with the target framework .NET CORE 3.1 and WindowsForms application types. The client submodule was written in HTML5 and JavaScript with the help of WebRTC-adapter library [15]. The server submodule uses the WebRTC Native API [16] while the first module, that does the temperature reading and detections, was written with the help of the SciShap.TensorFlow.NET, SciSharp.Keras.NET [17] and EmguCV .NET wrapper for OpenCV [18] packages.

The EmoTemp Android application runs without any dependencies. This is not the case for the desktop application, as it requires .NET CORE Runtime 3.1.4 [19], Python 3.7 with TensorFlow and Keras modules installed and a 64-bit OS. The minimum required hardware for the desktop application is a CPU like Intel's i5 7th generation with 4 Cores at 3.4 GHz or similar and 4 GB RAM. For the client submodule, the only dependency needed is a machine capable of running Google Chrome's web browser.

Flir One Pro camera has a thermal resolution of 160 x 120 pixels at a frame rate of 8.7 Hz, a visual resolution of 1440 x 1080 pixels and a thermal sensitivity of 0.07 Celsius degrees.

Flir A320 thermal camera has an infrared resolution of 320 x 240 pixels with a thermal sensitivity of 0.05 Celsius degrees with a frame rate Ethernet streaming of 4.5 Hz.

V. Limitations and Implications for Practice

For the moment, the developed embedded system faces few limitations:

- False alarms could be triggered if, due to the positiong of the cameras, in the FoV, some light spots coming from the Sun will be visible. These spots will generate heat, reaching temperatures of more than 40 Celsius degrees.

- The need to install on the server machine the software dependencies presented in the previous section
- Sometimes, upon the machine entering stand-by, the desktop application freezes and has to be restarted along with all the clients having to refresh the connection from the browser.
- For the desktop application, it is vital to have a static IP address allocated to the thermal camera so there will be no issues when there will be power cuts or internet connection drop offs. Otherwise, the camera will receive a different IP every time and the desktop application must be restarted to reflect the changes.
- Thermovision camera Flir One Pro has to be recharged after about two hours of usage.

In the following, we briefly present a few practical applications of our solution:

- The University's Management board has decided that the system will be implemented in other faculties also, especially where there will be a lot of students entering the faculty at once.
- As soon as the developed system idea reaches outside of University doors, public or private institues could request the system installation. For example, when entering a store, if the buyer does not have a mask, an alarm will be triggered specifying the need to put on a mask, along with the displayed temperature.
- Checking student's body temperature as a requirement for the participation in the exam's session, bachelor's degree or dissertation thesis exams at the entrance of the university and to verify if they are wearing a mask or not.

VI. Conclusions and Further Work

Based on two types of thermal vision cameras (fixed and mobile) we developed an embedded system (hardware-software) which identifies possible COVID-19 suspects by real time temperature screening and sending alarm signals over network or by SMS to local authorities. The system is capable, besides the human body temperature reading to also detect the age, gender, emotion and if the person is wearing a mask or not, using a smartphone. The software applications allows automatic finding of the top 5 highest temperature values from the current displayed frame. Basically the application analyzes and displayes frames in real time. The applications included a spinner that allows the user to set the threshold warning temperature value. If at least one out of the 5 highest temperature values found before is above the threshold, a warning signal will be sent and these values will be marked with red. We studied how the emissivity influences the reading of the temperature and found that the emissivity value has to be set to 0.98 for the most accurate results. We also studied how the distance between the camera and subject influences the temperature readings and our conclusion is that the temperature sensors of cameras are not sensitive at less than two meters providing the same temperature both at 20cm and 200cm.

For the future development we intend to implement a centralized database in order to visualize when (and where, because we will extend the project to other faculties) the infection did take place and also to improve the detection accuracies based on many images, features and algorithms. Our further intention is to analyze how a subject standing in light/shadow influences the temperature reading in case of the mobile camera scenario.

Acknowledgment

This work was partially developed under the project financed by Knowledge Transfer Center Hasso Plattner Institute at Lucian Blaga University of Sibiu.

References

[1] World Health Organization, "Laboratory testing for 2019 novel coronavirus (2019-nCoV) in suspected human cases," vol. 2019, no. January, pp. 1–7, 2020.

[2] J. R. Hageman, "The Coronavirus Disease 2019 (COVID-19)", Pediatr. Ann., vol. 49, no. 3, pp. e99–e100, 2020

[3] Francisco J. Sanchez-Marin, Sergio Carrera, and Carlos Villaseñor-Mora "Novel approach to assess the emissivity of the human skin," Journal of Biomedical Optics 14(2), 024006 (1 March 2009).

[4] Jones, Michael, and Paul Viola. "Fast multi-view face detection." Mitsubishi Electric Research Lab TR-20003-96 3.14 (2003):2.

[5] Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size." arXiv preprint arXiv:1602.07360 (2016).

[6] Eran Eidinger, Roee Enbar, and Tal Hassner, Age and Gender Estimation of Unfiltered Faces, Transactions on Information Forensics and Security (IEEE-TIFS), SI on Facial Biometrics in the Wild, Volume 9, Issue 12, pp. 2170 - 2179, Dec. 2014.

[7] Gil Levi and Tal Hassner, Age and Gender Classification Using Convolutional Neural Networks. IEEE Workshop on Analysis and Modeling of Faces and Gestures (AMFG), at the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Boston, 2015.

[8] Sandler, Mark, et al. "Mobilenetv2: Inverted residuals and linear bottlenecks.", Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[9] Kazemi, V., & Sullivan, J. (2014). "One millisecond face alignment with an ensemble of regression trees," at the IEEE Conf. on Computer Vision and Pattern Recognition (pp. 1867-1874).

[10] Rosebrok, Adrian "COVID-19: Face Mask Detector with OpenCV, Keras/TensorFlow, and Deep Learning" https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/ May 4 2020. Accessed 24th May 2020.

[11] E.Lundqvist, D., Flykt, A., & Öhman, A. (1998). The Karolinska Directed Emotional Faces - KDEF, CD ROM from Department of Clinical Neuroscience, Psychology section, Karolinska Institutet.

[12] P. Lucey, J. F. Cohn, T. Kanade, J. Saragih, Z. Ambadar and I. Matthews, "The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression," 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, San Francisco, CA, 2010, pp. 94-101, doi: 10.1109/CVPRW.2010.5543262.

[13] OpenCV4Android versions 3.4.3 https://opencv.org/android/. Accessed 24th May 2020.

[14] TensorFlowLite for Android https://www.tensorflow.org/lite/guide/android. Accessed 24th May 2020.

[15] WebRTC adapter https://github.com/webrtc/adapter . Accessed 24th May 2020.

[16] WebRTC Native http://webrtc.github.io/webrtc-org/native-code/native-apis . Accessed 24th May 2020.

[17] SciSharp.TensorFlow.NET & SciSharp.Keras.NET https://github.com/SciSharp. Accessed 24th May 2020.

[18] EmguCV .NET Platform http://www.emgu.com/wiki/index.php/Main_Page . Accessed 24th May 2020.

[19] DotNet Core 3.1 SDK https://dotnet.microsoft.com/download/dotnet-core/3.1. Accessed 24th May 2020.