UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

On the use of a low-cost embedded system for face detection and recognition

Ramiro Aleksey Sandoval Avakimova
Vanessa Fernanda Camino Guerra

Ingeniería en Sistemas

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniería en Sistemas

Quito, 4 de mayo de 2020

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN
DE TRABAJO DE FIN DE CARRERA

On the use of a low-cost embedded system for face detection and face recognition

Vanessa Fernanda Camino Guerra

Nombre del profesor, Título académico **Noel Pérez Pérez, Ph.D**

Quito, 4 de mayo de 2020

DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Nombres y apellidos: Ramiro Aleksey Sandoval Avakimova

Código: 00130256

Cédula de identidad: 1719116145

Lugar y fecha: Quito, mayo de 2020

Nombres y apellidos: Vanessa Fernanda Camino Guerra

Código: 00130814

Cédula de identidad: 1722822721

Lugar y fecha: Quito, mayo de 2020

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

RESUMEN

Este documento explora la viabilidad de usar componentes disponibles comercialmente para implementar un sistema embebido de bajo costo como el núcleo de un sistema de detección y reconocimiento facial. El sistema está compuesto por un módulo de cámara para Raspberry Pi y un Raspberry Pi B+, potenciado por un Intel Neural Compute Stick 2. Cuatro modelos de aprendizaje supervisado fueron implementados en el sistema embebido para el reconocimiento facial bajo diferentes condiciones con el objetivo de determinar las limitaciones y capacidades del sistema, y las mejores condiciones de funcionamiento. Los mejores resultados fueron obtenidos usando el algoritmo del Perceptron Multicapa (MLP), cuando el individuo se encontraba a una distancia entre 0.3 a 1 metros de la cámara, el factor de iluminación en el rango de 115 a 130 lux y la rotación horizontal de la cara entre -5° a +5°.

Palabras clave: detección facial, reconocimiento facial, sistema embebido, Raspberry pi 3 b+, Intel neural stick 2, Caffe, OpenFace.

ABSTRACT

This paper explores the feasibility of using commercially available off-the-shelf components to implement a low-cost embedded system as the core of a facial detection and recognition system. The system is composed of a Raspberry Pi camera module and a Raspberry Pi B+ enhanced by an Intel Neural Compute Stick 2. Four supervised learning models were implemented on the embedded system for face recognition under different conditions to determine the limitations and capabilities of the system, and the best operational conditions. Best results were achieved when using a Multilayer Perceptron (MLP) algorithm and the distance of the subject to the camera was between 0.3 to 1 meters, the illumination factor in the range from 115 to 130 lux and the horizontal face rotation between -5° to +5°.

Keywords: face detection, face recognition, embedded system, Raspberry pi 3 b+, Intel neural stick 2, Caffe, OpenFace.

TABLE OF CONTENTS

TABLE INDEX

FIGURE INDEX

1. INTRODUCTION

Face detection and recognition can be considered well-known topics as they have been studied for several years. Different robust computer vision algorithms or techniques have been proposed in order to provide better levels of accuracy while being computationally effective. As a consequence, there is a plethora of options to be considered when a face detection and recognition system must be designed from scratch. According to Yang et al. (2002) face detection tackles different challenges regarding to pose, presence or absence of structural components, facial expressions, occlusion, image orientation and image conditions.

On the other hand, as mentioned by Ehsan et al. (2015), image processing and computer vision algorithms are generally computation and data intensive in nature. In this regard, a face detection and recognition system should be able to strike a trade-off between the effectiveness and the cost. Thus, the choice of a certain computer vision algorithm based on the available hardware do not represent a trivial exercise at all.

Nowadays, face detection and recognition are used in different comprehensive systems. For instance, door access control, surveillance and suspect detection, face-based bio-metrics, wearables for law enforcement, among others (Mahmood et al., 2017).

These systems try to use the latest advances in computer vision and artificial intelligence while minimizing the cost. According to the state-of-the-art, embedded systems are the preferred option to implement face detection and recognition procedures. An embedded system should include sufficient enough computational resources required to execute computer vision algorithms at an affordable cost.

In the present work, the embedded-based face detection and recognition concept is further analysed to determine limitations when using off-the-shelf components to maintain the overall price of the low-cost solution. Regarding face detection, convolutional architecture for fast feature embedding (Caffe) framework (Jia et al., 2014) is used. This framework includes a face detector that provides a better performance in comparison with traditional face detection algorithms such as Viola-Jones (Granger et al., 2017). Regarding the use of artificial intelligence techniques to carry out face recognition, most of proposals follow an off-the-shelf based approach.

In traditional systems, the embedded system is in charge of capturing the image, detecting the face, performing a pre-processing and finally executing the feature extraction process. In the next step, the extracted features are sent to a remote server to perform the recognition process. Although this approach leverages the computational power of remote servers, the communication channel between the embedded system and the remote server represents a single point of failure. If the channel is unavailable, the system is unable to identify a person. Besides, the delay that the communication channel adds might restrict the real-time responsiveness of the face recognition process. In this sense, an Intel compute neural stick was used in our approach to provide additional computational resources that a comprehensive embedded system requires. On the other hand, the single point of failure is suppressed and the real-time responsiveness of the system is guaranteed.

Apart from the particularities of the architecture of the proposed embedded system, this work presents an strategy to evaluate and chose multiple face recognition classifiers trained under several hyper-parameters and a set of experimental analysis that test the limits of such classifiers under different distance, illumination and face rotations conditions. In particular, traditional well-known classification algorithms were tested, i.e., support vector machines (SVM), artificial neural networks (ANN), k-nearest neighbors (KNN), and random forest (RF). An exhaustive search strategy was used to select optimum hyper-parameters and the best models were tested and ported to the embedded system. Accuracy and area under the receiver operating characteristic curve (AUC) were used for the evaluation.

The remainder of the article is structured as follows. In section 2, the state of the art is analyzed and the particular features of the proposed approach are remarked. In section 3, the components of the proposed embedded system are described. In section 4, the methodology employed to build the detection and recognition system is introduced. To better understand this approach and to validate this proposal, several tests with the corresponding analysis are presented in section 5. Finally, the conclusions of the paper and future works are reported (see section 6).

2. RELATED WORK

Several works propose the use of embedded systems to implement face detection and recognition procedures due to the low cost associated with them. For instance, Sajjad et al. (2019) propose a facial expression recognition framework for law-enforcement services. The framework uses a Raspberry Pi and follows a cloud-assisted approach. Regarding the methodology, Viola-Jones is used as face detector and fast rotated BRIEF (ORB) descriptor is used for features extraction. Then, the feature vector is sent to an SVM-based multi-classifier for face recognition. Although Viola-Jones provides effective results, Granger et al. (2017) state that Caffe framework provides a region convolutional neural network (R-CNN) based face detector that outperforms Viola-Jones. On the other hand, as mentioned earlier, the cloud-assisted approach represents a single point of failure that might restricts the real-time responsiveness of the system.

A real-time emotional state detection using a Field-programmable gate array (FPGA) is proposed by Turabzadeh et al. (2017). A uniform local binary patterns (LBP) algorithm is used for image feature extraction. LBP is in charge of providing a single features vector that describes the entire image. Regarding the classification, KNN regression algorithm is used. Although LBP performs well under high illumination variability, the accuracy might be worsened due to features redundancy. On the other hand, while it is true that an FPGA is able to provide a better performance, the flexibility to use different open source image processing and machine learning applications is restricted.

Chen et al. (2016) propose a low-cost face recognition system. Extended LBP, principal component analysis (PCA) and sparse representation (SRC) are used for face recognition while Viola-Jones is used as face detector. Regarding SRC, Zhang et al. (2015) state that the effectiveness and efficiency of sparse representation methods cannot perfectly meet the requirements of real-world applications. Random corruptions, varying illumination, outliers, occlusion and complex backgrounds might restrict the robustness and performance of SRC.

A computer network based face detection and recognition system is proposed by Wazwaz et al. (2018). Raspberry Pi executes the boosted cascade of simple features (BCOSF)

algorithm (Viola-Jones) as face detector. In order to perform face detection and recognition, a cluster of remote servers executes the local binary pattern histograms (LBPH) algorithm. It should be noted that the reliability as well as the real-time responsiveness of the system rely on the computer network.

3. PROPOSED EMBEDDED SYSTEM

With the aim of providing an unified detection and recognition system, an embedded platform that integrates specific hardware and third-party libraries, is proposed as depicted in Fig. 1. Besides, a light-software layer has been developed in order to manage the different hardware and software components. The functional parts of the proposed system are explained next.

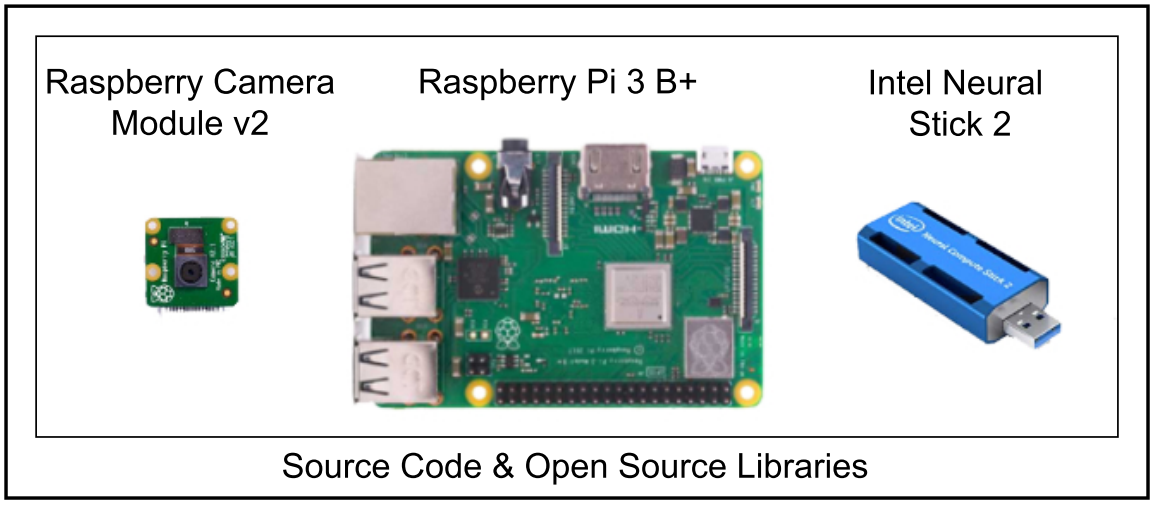

Figure 1

Proposed embedded system

3.1 Hardware

The embedded system is composed of a Raspberry Pi 3 B+ and an Intel Neural Stick 2 (INCS2) (Corporation, n.d.). The INCS2 is a dedicated deep neural network hardware accelerator for computer vision and AI inference. The Raspberry is equipped with a Cortex-A53 (ARMv8) 64-bit processor and 1GB of SDRAM (Foundation, n.d.-b). A 8 MP camera module is connected to the CSI Camera Port of the Raspberry Pi (Foundation, n.d.-a). The latest available version of Raspbian is used as the Operating System on the Raspberry.

3.2 Third-party source code

In order to create appropriate data structures as well as to train, evaluate and generate different face recognition algorithms, several open source Python libraries are incorporated.

For instance: scikit-learn v0.22.1: a simple and efficient open source library for data analysis (Pedregosa et al., 2011). Opencv-python v4.1.2.30: a library containing prebuilt opencv packages (Itseez, 2015). Pickle: implements binary protocols for serializing and de-serializing Python object structures (Van Rossum & Drake, 2009). Caffe precompiled Face Detector: (Convolutional Architecture for Fast Feature Embedding) is a deep learning framework (Jia et al., 2014). OpenFace precompiled embedder: an implementation of face recognition with deep neural networks (Amos et al., 2016).

4. METHODOLOGY

4.1 Face database

The face database used in training and experimental evaluation is composed of 450 images. The database includes thirty images per individual from fifteen different individuals. Photos were taken at the same place, under artificial light, with a white background. The face database was created using a Logitech C920 full HD camera connected to a laptop PC. The camera was placed at a height of 1.20 m from the ground and 1.0 m away from the individual. Photos were saved in 720x720 pixels size. Participants were asked to do different facial expressions and to look at the camera from different angles while they were sitting.

4.2 Model configuration

A block diagram of the system's software operation is shown in Fig. 2. Image pre-processing for training and training itself was not done in the embedded system. The training process and the face detection and recognition procedures performed by the proposed system are described next.

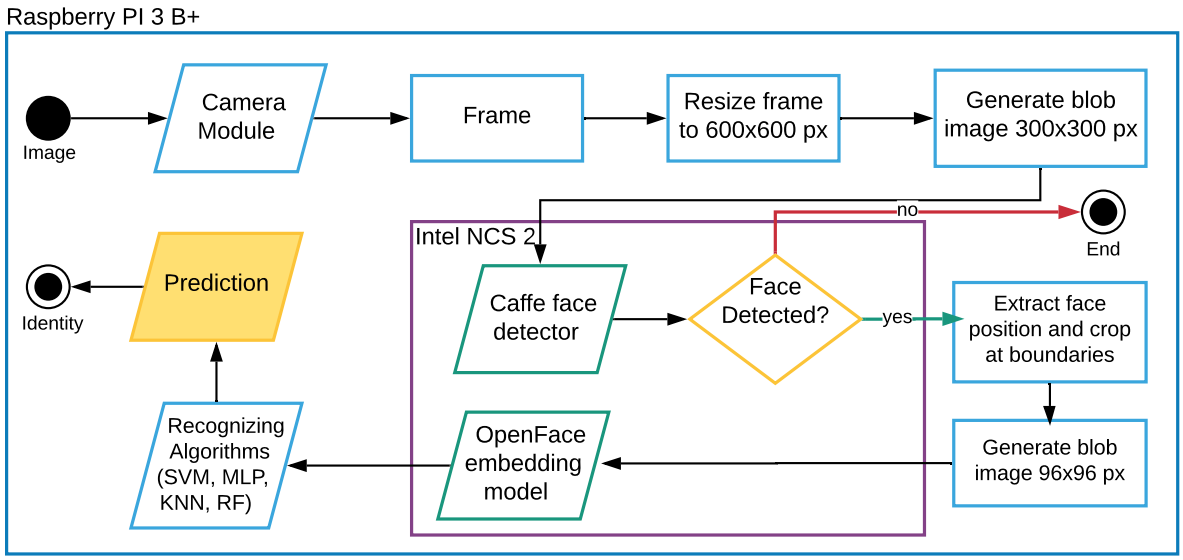

Figure 2

Block diagram of the system

4.2.1 Image preprocessing.

Each image is resized to 600x600 pixels. After the resizing process, using the Deep Neural Network OpenCV library (DNN), a 300x300 pixels blob image is obtained. This blob image is processed by a precompiled face detector provided by Caffe Framework and for model training. Caffe's precompiled model is selected as it is fast and effective face detector that relies on a Region Convolution Neural Network (R-CNN) (Jia et al., 2014). After the face detection procedure, the dimensions and position of the region of interest are stored. This region is cropped at boundaries and resized to 96x96 pixels. This image is then used as input for a precompiled embedding model provided by OpenFace. This embedding model combines dlib's real-time pose estimator and OpenCV's affine transformation to standardize the alignment of the outer eyes and nose for all images and to obtain a 128-dimension representation of the image (Amos et al., 2016). The described image representation methodology is used to train the different recognition algorithms during the test and real-time implementation.

4.2.2 Model training.

Four supervised learning algorithms were evaluated. For the experimental evaluation, the sklearn's implementation of SVM, Multilayer Perceptron (MLP), KNN and RF algorithms is used. Training and test sets are generated from the embedded images by using the K-fold cross validation (Anguita et al., 2012) methodology. A grid search (GS) (Syarif et al., 2016) customized implementation is used to find the best hyper-parameters for the evaluated algorithms. GS exhaustively tries multiple combinations of hyper-parameters by performing cross product between an user-provided list of parameters. Then, prediction results are compared between the generated models. At the end, GS returns the list of hyper-parameters from the best performer model.

SVM (Yu & Kim, 2012) is a supervised learning algorithm capable of performing binary and non lineal classification. In this case, the non lineal implementation is used to map the possible outputs, represented by hyper-planes, to a high-dimensional feature space. Unseen input images are classified in one of the possible hyper-planes. Possible tunable

hyper-parameters in the sklearn's implementation that were explored are: the *kernel*, regularization parameter *C*, and *gamma*.

MLP (Noriega, 2005) is a class of feed forward neural network that uses back-propagation for training. When the input is processed, the error of the output and the expected output is measured using the least minimum squares technique. The weights of the network are recalculated to minimize the error. Possible tunable hyper-parameters in the sklearn's implementation that were explored are: the *hidden layer sizes*, *activation function*, *solver* and the *learning rate*.

KNN (Cover & Hart, 1967) is a supervised learning algorithm used for classification or regression. At the training stage, feature vectors and its corresponding label are stored. When an unseen sample is tested, classification is done by placing the sample's vector in the feature space and assigning the predominant label of its *k* nearest neighbors. Nearest neighbors are selected by following a weight assignment, usually the Euclidean distance. Possible tunable hyper-parameters in the sklearn's implementation that were explored are: number of neighbors *k* and *weight* calculation methodology.

RF (Breiman, 2001) is a classification algorithm based on the exploration of multiple generated Decision Trees (DTs), that work as an ensamble. During training, random samples are selected to train the DTs. Predictions for unseen samples are obtained by averaging the result obtained from each DT. Possible tunable hyper-parameters in the sklearn's implementation that were explored are: number of DTs *n*, function to measure splits *criterion* and methodology to determine the maximum number of features for each tree split *max_split*.

4.2.3 Real time detection and recognition.

The model generated in the PC was exported using the pickle library, which dumps the whole python object to a file. This file was then used to import the model into the embedded system using the same library.

Once the model is imported to the Raspberry, OpenCV library is used to specify the Intel Neural Compute Stick 2 as the default backend.The Stick executes the preprocessing methodology for every frame captured by the camera in real-time. Then, this preprocessed frame is delivered to the active learning algorithm. Finally, by using the built-in functions of

the implemented algorithm, an output label is assigned to the face area in real-time as shown in Fig 3. This label contains the recognized face id and the corresponding confidence level.

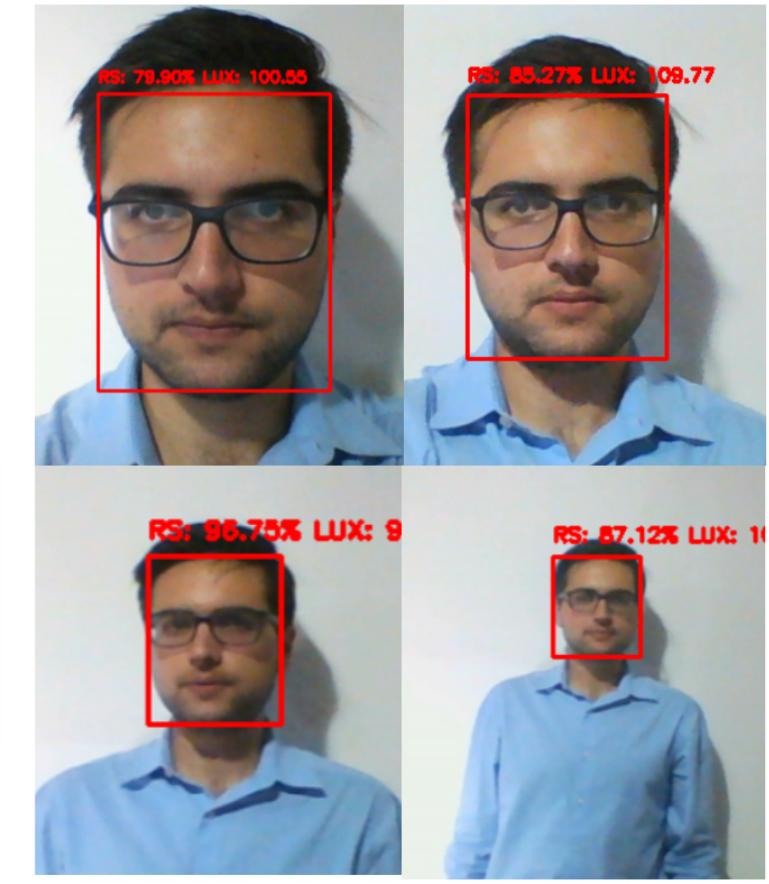

Figure 3

Example of a successful real-time recognition of an individual (labeled as RS) under the MLP algorithm at different distances (0.3, 0.5, 1.0 and 2.0 meters from the camera)

4.3 Validation metrics

Validation metrics are obtained by comparing the output generated from perceiving an individual by the camera at different conditions in terms of distance, lighting and rotation conditions to the expected result. The output label and level of confidence are obtained from sending the image to the trained model and predicting the output with the built-in functions in the sklearn's library for each algorithm. The results obtained are shown in Tables 2, 3 and 4 and Fig. 5.

5. RESULTS AND DISCUSSION

5.1 Training

The list of hyper-parameters from the best performer models, with their respective accuracy results as well as ROC-AUC metrics, obtained during the training stage are shown in Table 1.

Table 1

Accuracy and ROC-AUC metrics for the training stage for different supervised learning algorithms

Algorithm	**Best Hyper-parameters**	**Test Set Score (%)**	**ROC-AUC Score (%)**
SVM	'C': 100.0, 'gamma': 'auto', 'kernel': 'rbf'	89	97
MLP	'hidden_layer_sizes': (100,), 'activation': 'relu', 'solver': 'adam'	91	100
KNN	'n_neighbors': 13, 'weights': 'distance'	79	91
RF	'n_estimators': 500, 'criterion': 'entropy', 'max_features': 'log2'	89	95

5.2 Effectiveness of the proposed system

An example of how the system works is shown in Fig. 3, the system is also capable of performing multiple individual face detection and recognition as shown in Fig. 4. These images show how the system detects the face, wraps it within a red rectangle and shows an output label with the level of confidence for the recognition and the amount of illumination perceived on the face region. However, the performance of this system is bounded to illumination conditions, face rotations and the distance of the individual from the camera, as analysed next. The average results obtained from the results of 30 recognizing tests using different MLCs on two individuals at different distances, lightning conditions and horizontal face rotations are highlighted in Tables 2, 3, and 4.

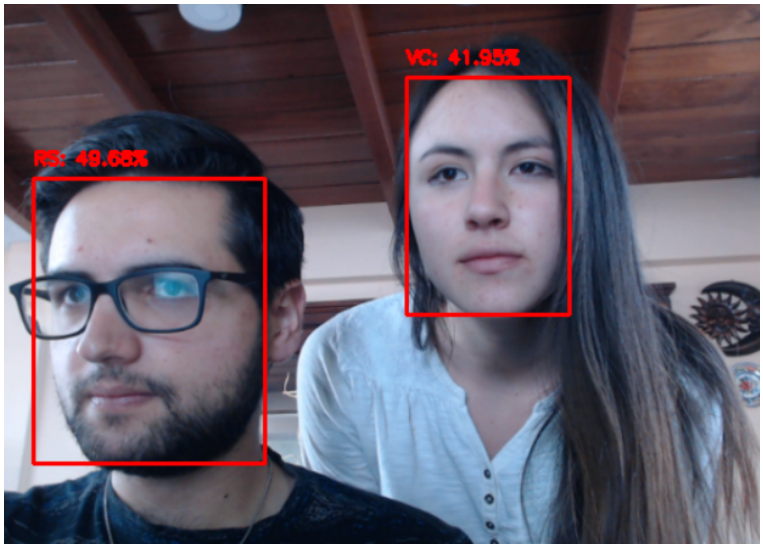

Figure 4

Example of a successful real-time recognition of two individuals at the same time (labeled as RS and VC, respectively) under the SVM algorithm

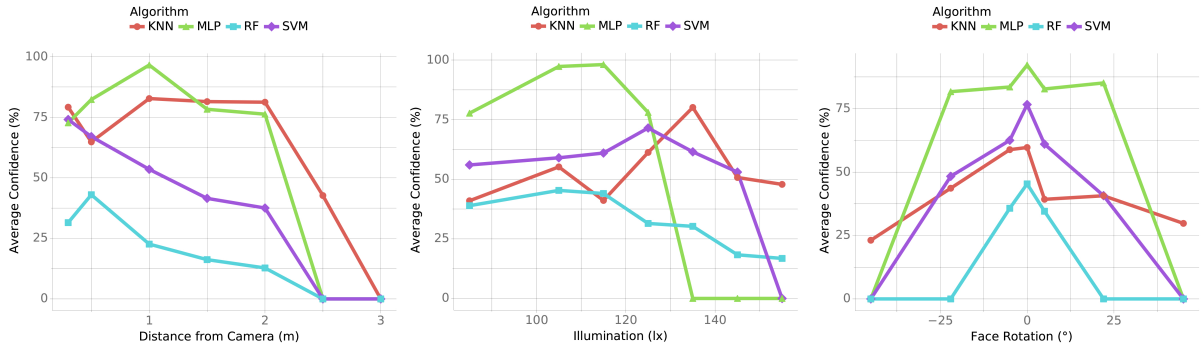

Figure 5

A comparison of real-time performance between algorithms under different distance from camera, illumination conditions and horizontal face rotation

5.2.1 Distance analysis.

The recognition system was tested for different distances (between 0.3 to 3 m) to determine the best range of operation (see Fig. 5 and Table 2). The system was unable to detect faces when individuals are at a distance greater than or equal to 3.0 m. Effectiveness of recognition decreases as the distance between the test subject and the camera increases. Best recognition results were obtained when the test subject was between 0.3 to 1.5 m away from the camera.

Table 2

Real-time average confidence of recognition (%) vs Distance (m) under different algorithms

Distance from Camera (m)	**SVM Average Confidence (%)**	**MLP Average Confidence (%)**	**KNN Average Confidence (%)**	**RF Average Confidence (%)**
0.3	74.08	72.70	79.17	31.50
0.5	67.02	82.32	64.83	43.02
1.0	53.51	96.57	82.73	22.63
1.5	41.52	78.31	81.47	16.22
2.0	37.53	76.28	81.23	12.78
2.5	Not Detected	Not Detected	42.68	Wrong Output
3.0	Not Detected	Not Detected	Not Detected	Not Detected

5.2.2 Illumination analysis.

Here, illumination limitations for the system are tested. A frontal artificial light was placed at different intensities at the recommended distance of 0.5 m from the camera. Illumination, in lux, was measured by calculating the pixel intensities in the face detected region. According to Table 3, when illumination was greater than 150 lux, the prediction was no longer accurate and it also affected face detection. Better recognition results were obtained when illumination conditions were between 115 and 130 lux (see Fig. 5).

Table 3

Real-time average confidence of recognition (%) vs Frontal Illumination (lx). under different algorithms

Frontal Illumination (lx)	SVM Average Confidence (%)	MLP Average Confidence (%)	KNN Average Confidence (%)	RF Average Confidence (%)
85	56	77.7	40.98	38.92
105	59	97.3	55.19	45.33
115	61	98.1	41.1	44.02
125	71.5	78.0	61.25	31.46
135	61.5	Wrong Output	80.12	30.25
145	53.0	Wrong Output	50.74	18.31
155	Wrong Output	Wrong Output	47.85	16.79

5.2.3 Horizontal face rotation analysis.

Horizontal facial rotations were also considered to evaluate the system. Facial rotations were done at the recommended distance of 0.5 m from the camera and with artificial lightning conditions of 115 lux. Horizontal face rotations were measured by drawing on the wall points representing different rotations, in degrees, from the center position. Test subjects were asked to align their nose with each wall point at a time. As shown in Table 4, the level of confidence of the recognition tends to decrease as the rotation increases. Also, the system was only able to detect faces when the face rotation angle was less than 45 degrees from center position. A frontal approach or a rotation between -5 or 5 degrees is therefore recommended to obtain better results (see Fig. 5).

5.2.4 Face accessories analysis.

The system was unable to recognize correctly after changes in face accessories. For example, at the recommended ranges described above, if someone wore glasses at the moment of creating the database and then, the person stops wearing them, the subject was no longer

Table 4

Real-time average confidence of recognition (%) vs Horizontal Face Rotation (deg.) under different algorithms

Horizontal Face Rotation (deg.)	SVM Average Confidence (%)	MLP Average Confidence (%)	KNN Average Confidence (%)	RF Average Confidence (%)
+0	76.6	92.17	59.68	45.33
-5	62.5	83.52	58.82	35.72
+5	61	82.74	39.25	34.55
-22	48.3	81.67	43.62	Wrong Output
+22	40.7	85.12	40.65	Wrong Output
-45	Wrong Output	Wrong Output	23.12	Wrong Output
+45	Wrong Output	Wrong Output	29.77	Wrong Output

recognizable by the system. This also applies to the use of sunglasses. Other accessories were not explored.

The embedded system perceives about 2.11 frames per second (FPS) on average between algorithms. FPS in this system means how many times the whole image preprocessing, detection and recognition can be done for each second. For comparison, the same code was executed on a desktop class computer and the embedded system without the INCS2 under ideal conditions. The desktop system perceived 11.7 FPS on average and the Raspberry without the INCS2 1.89 FPS. That represents an increase in frames processed per second of 11.64% when the INCS2 is used. Since the model in both the Raspberry and the desktop are the same, there is no statistical difference in the output labels and level of confidence obtained between systems, for comparison refer to Table 5.

5.3 Limitations

Factors such as distance from the camera, illumination conditions, rotation of the face, and face accessories changes affect the recognition accuracy. Therefore, there are specific and recommended working ranges for each of the described factors. The size of the trained model

Table 5

Average FPS perceived and average accuracy under ideal conditions between Desktop and the Embedded System with and without Intel Neural Stick 2 under the SVM algorithm

Metric	**Desktop**	**Raspberry PI 3 B+**	
		With INCS2	**Without INCS2**
FPS	11.73	2.11	1.89
Confidence (%)	74.32	74.06	74.24

increases with the number of images used in the training phase and the number of individuals to recognize. As described in the Hardware subsections, the Raspberry 3 B+ has only 1 GB of RAM, so it relies on memory swapping to be able to recognize in real time, downgrading the performance. Another important aspect is that the hardware interface and bus speeds of the Raspberry limits the FPS perceived by the camera, also downgrading the performance. It's worth to mention that the embedded camera resolution is not good enough to identify individuals at long range.

6. CONCLUSIONS AND FUTURE WORK

In this work, the performance of a proposed embedded system as the core of a facial recognition system is evaluated. Recommended operation ranges were determined for this system and are detailed next. The best range of operation for this system was between 0.3 to 1 m. For the illumination factor, the recommended range is between 115 and 130 lux, while the system was only capable of detecting horizontal face rotation between -5 to +5. Best real-time results were obtained under the MLP algorithm. The performance of the system is better when it is tested under similar conditions to those had when capturing the training photos. It is recommended to create the face database in similar operation distance, lightning conditions and facial rotations to the final use area. In general, results and performance obtained under optimal distance, face rotations and lighting were good despite the computational limitations of this low-cost system.

As part of our future work, another face database will be constructed in which multiple conditions of lightning, facial expressions and distances for each of the subjects will be taken into consideration. Finally, more powerful systems that can be bought for a similar price (this system can be acquired for about $150,00 USD), such as the ODROID-N2 or the new Raspberry Pi 4 B should be interesting hardware to test as the core of the facial detection and recognition system. We expect that as new hardware with more computational power come available the hardware limitations reported in this study will be overcome.

7. REFERENCES

Amos, B., Ludwiczuk, B., & Satyanarayanan, M. (2016). *Openface: A general-purpose face recognition library with mobile applications* (tech. rep.). CMU-CS-16-118, CMU School of Computer Science.

Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., & Ridella, S. (2012). The'k'in k-fold cross validation., In *Esann*.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Chen, Y.-P., Chen, Q.-H., Chou, K.-Y., & Wu, R.-H. (2016). Low-cost face recognition system based on extended local binary pattern, In *2016 international automatic control conference (cacs)*. IEEE.

Corporation, I. (n.d.). Intel® neural compute stick 2 (intel® ncs2).

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE transactions on information theory*, *13*(1), 21–27.

Ehsan, S., Clark, A. F., Rehman, N. U., & McDonald-Maier, K. D. (2015). Integral images: Efficient algorithms for their computation and storage in resource-constrained embedded vision systems. *Sensors*, *15*(7), 16804–16830.

Foundation, R. P. (n.d.-a). Raspberry camera module v2.

Foundation, R. P. (n.d.-b). Raspberry pi 3 model b+.

Granger, E., Kiran, M., Blais-Morin, L.-A., Et al. (2017). A comparison of cnn-based face and head detectors for real-time video surveillance applications, In *2017 seventh international conference on image processing theory, tools and applications (ipta)*. IEEE.

Itseez. (2015). Open source computer vision library.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., & Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding, In *Proceedings of the 22nd acm international conference on multimedia*, Orlando, Florida, USA, Association for Computing Machinery. https://doi.org/10.1145/2647868.2654889

Mahmood, Z., Muhammad, N., Bibi, N., & Ali, T. (2017). A review on state-of-the-art face recognition approaches. *Fractals*, *25*(02), 1750025.

Noriega, L. (2005). Multilayer perceptron tutorial. *School of Computing. Staffordshire University*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Sajjad, M., Nasir, M., Ullah, F. U. M., Muhammad, K., Sangaiah, A. K., & Baik, S. W. (2019). Raspberry pi assisted facial expression recognition framework for smart security in law-enforcement services. *Information Sciences*, *479*, 416–431.

Syarif, I., Prugel-Bennett, A., & Wills, G. (2016). Svm parameter optimization using grid search and genetic algorithm to improve classification performance. *Telkomnika*, *14*(4), 1502.

Turabzadeh, S., Meng, H., Swash, R. M., Pleva, M., & Juhar, J. (2017). Real-time emotional state detection from facial expression on embedded devices, In *2017 seventh international conference on innovative computing technology (intech)*. IEEE.

Van Rossum, G., & Drake, F. L. (2009). *Python 3 reference manual*. Scotts Valley, CA, CreateSpace.

Wazwaz, A. A., Herbawi, A. O., Teeti, M. J., & Hmeed, S. Y. (2018). Raspberry pi and computers-based face detection and recognition system, In *2018 4th international conference on computer and technology applications (iccta)*. IEEE.

Yang, M.-H., Kriegman, D. J., & Ahuja, N. (2002). Detecting faces in images: A survey. *IEEE Transactions on pattern analysis and machine intelligence*, *24*(1), 34–58.

Yu, H., & Kim, S. (2012). Svm tutorial-classification, regression and ranking. *Handbook of Natural computing*, *1*, 479–506.

Zhang, Z., Xu, Y., Yang, J., Li, X., & Zhang, D. (2015). A survey of sparse representation: Algorithms and applications. *IEEE access*, *3*, 490–530.