Facial-based Intrusion Detection System with Deep Learning in Embedded Devices

Giuseppe Amato, Fabio Carrara, Fabrizio Falchi, Claudio Gennaro, Claudio Vairo
Institute of Information Science and Technologies of the National Research Council of Italy (ISTI-CNR),
via G. Moruzzi 1, 56124, Pisa, Italy
giuseppe.amato@isti.cnr.it,fabio.carrara@isti.cnr.it,fabrizio.falchi@isti.cnr.it,
claudio.gennaro@isti.cnr.it,claudio.vairo@isti.cnr.it

ABSTRACT

With the advent of deep learning based methods, facial recognition algorithms have become more effective and efficient. However, these algorithms have usually the disadvantage of requiring the use of dedicated hardware devices, such as graphical processing units (GPUs), which pose restrictions on their usage on embedded devices with limited computational power.

In this paper, we present an approach that allows building an intrusion detection system, based on face recognition, running on embedded devices. It relies on deep learning techniques and does not exploit the GPUs. Face recognition is performed using a *k*nn classifier on features extracted from a 50-layers Residual Network (ResNet-50) trained on the VGGFace2 dataset. In our experiment, we determined the optimal confidence threshold that allows distinguishing legitimate users from intruders.

In order to validate the proposed system, we created a ground truth composed of 15,393 images of faces and 44 identities, captured by two smart cameras placed in two different offices, in a test period of six months. We show that the obtained results are good both from the efficiency and effectiveness perspective.

CCS CONCEPTS

• **Security and privacy→Intrusion detection systems** • **Computing methodologies→Computer vision tasks** • **Computer systems organization→Embedded systems**

KEYWORDS

Intrusion detection, Facial recognition, Deep learning, Convolutional Neural Network, Embedded devices

1 INTRODUCTION

Facial recognition is an important task in security and surveillance. In this paper, we present an intrusion detection system for embedded devices that is based on facial recognition. Our system is composed of smart cameras (i.e. video cameras capable of processing and analyzing the acquired data) to monitor the access to restricted area. The proposed solution is based on a set of authorized persons' faces and it exploits a facial recognition algorithm to determine if the person entering the monitored environment belongs to the set of authorized people or not.

Several approaches have been proposed in the last few years to implement the face recognition task. LBP [1, 2] combines local descriptors of the face in order to build a global representation that can be used to measure the distance with other LBP features.

Recently, Deep Learning and Convolutional Neural Network (CNN) approaches have been proposed that highly improved the performance in executing the face recognition task. However, these approaches usually require high computational power and dedicated GPUs to work and to achieve good results [11, 23, 26].

In this paper, we present a system for monitoring the access to a restricted area in a building, like office rooms, by using a facial recognition algorithm implemented with a Deep Learning approach. Our solution is aimed to embedded platforms that do not exploit the computational power of the GPUs. In particular, we deployed our system on a Raspberry Pi and we validated our approach by performing an accuracy evaluation on a face dataset of 15,393 faces belonging to 44 different identities that we built in our facilities. We also made publicly available the software we produced.

The rest of the paper is organized as follows. Section 2 presents some related work. In Section 3, we describe the proposed approach and the technique used to perform the facial recognition task. Section 4 gives an overview of the implemented system. In Section 5, we report the evaluation results that validate our approach. Finally, Section 6 concludes the paper.

2 RELATED WORK

The use of face information to recognize the identity of a person is a research area experiencing rapid development, thanks to recent advances in deep learning. Deep Learning [20] is a branch of Machine Learning that allows a neural network, composed of large number of layers, to learn representations of input data with increasing level of abstraction. Deep learning approaches provide near-human level accuracy in performing tasks like image classification [19], object detection [13], object recognition [9], sentiment analysis [28], speech recognition [12], parking monitoring [5, 6], face recognition [8, 23], and more. Deep features learned from CNNs have shown impressive performance in classification and recognition tasks. For instance, 99.77% of accuracy on the Labeled Faces in the Wild dataset [18] has been achieved by Liu et al. [21] and 99.33% by Schroff et al. [26].

In this work, we developed an embedded intrusion detection system based on a facial recognition algorithm that exploits deep

SSIP 2018, October 12–14, 2018, Prague, Czech Republic

ACM ISBN 978-1-4503-6620-5/18/10…$15.00
DOI: https://doi.org/10.1145/3290589.3290598

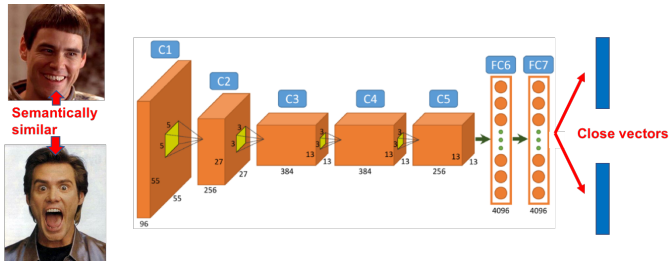

Figure 1: Extraction and usage of deep feature for facial recognition.

learning. In the literature, there are several works focusing on facial recognition in the embedded environment [3, 14, 24, 29, 30]. None of these uses deep learning for facial recognition, and do not even report the accuracy of the recognition, with the exception of [14], which however is limited since it has been carried out only on one authorized person and few unauthorized persons. The paper by Alippi et al. [4] aims at filling the gap between CNNs and embedded systems by introducing a methodology for the design and porting of CNNs to limited in resources embedded systems. In order to achieve this goal, however, they employed approximate computing techniques to reduce the computational load and memory occupation of the deep learning architecture by compromising accuracy with memory and computation. Moreover, the proposed methodology has been validated just on two well-known CNNs, i.e., AlexNet and VGG-16.

To the best of our knowledge, so far, the only work similar to ours that uses the features extracted from the network trained on VGGFace2 to implement facial recognition is [25]. This work obtained comparable values of facial recognition in unconstrained conditions (i.e., various illumination, pose, presence of noise, etc.) using a probabilistic approach to nearest neighbor classification, though at the cost of a higher computational complexity when compared to either baseline nearest neighbor or traditional SVM. We plan to make a more rigorous comparison in a future publication.

3 DEEP FEATURES

The approach used in this work to perform face recognition, and to determine if a person is authorized or not, is to use a k-Nearest Neighbor (kNN) [7] classification algorithm on facial features extracted using Deep Learning techniques.

Features are extracted using CNNs in a peculiar way. CNNs exploits a large amount of labeled input data to learn, in each internal layer, a higher-level representation of the input image, from low-level pixels to high-level global representation of the image. The output of the CNN is the predicted class, a face identity in case of facial recognition, coupled with its confidence value.

The classification prediction of these types of networks is very accurate. However, in a facial recognition scenario, in order to be able to use the prediction output of the CNN, the network needs to be trained on all the persons that need to be recognized. This requires a dataset composed of lots of images for each authorized person. If a new person needs to be added later, to the set of recognized faces, the network has to be retrained or fine-tuned. This is not always feasible, especially in a runtime system, where replacing a CNN on the fly is not easy to be done. On the other hand, we said that a CNN learns high-level representations of the input images in its internal layers. This information can be used as a global descriptor (feature) of the face. In particular, we used the output of the penultimate layer of the CNN, as a feature of the face. The use of internal layers to extract visual features (*deep features*) has been explored and assessed in various works, as for instance [10]. A nice property of deep features is that if the distance of two deep features vectors, computed from two different input images is small, we can assume that the two input images are semantically similar, and thus that they belong to the same person (see Figure 1). This information, once computed, can be stored and reused later to perform kNN classification.

In this paper, we used and assessed the performance of our approach in the facial recognition task by exploiting three CNN architectures fine-tuned on two different datasets, all provided by the Visual Geometry Group of the University of Oxford: a VGGNet [27] trained on the VGGFace dataset, and two ResNet models (ResNet-50 [16] and SE-ResNet-50 [17]) trained on the VGGFace2 dataset. All these CNNs are pre-trained to recognize people faces.

The VGGNet network used in [23] is composed of 16 convolutional layers and trained to recognize faces on a dataset composed of 2,6 million faces belonging to 2,6 thousand different persons. We took the output of the fully-connected 7th layer of the CNN (fc7) as deep feature. This is a 4096 size float vector, and it is the higher-level representation of the input face.

The VGGFace2 dataset [11] is the evolution of the VGGFace one. In our work, we used the two proposed models trained on VGGFace2, which are the ResNet-50 CNN (shortly ResNet-50_ft) and the ResNet-50 CNN with Squeeze-and-Excitation blocks (shortly SENet_ft). Both models are trained on the MS-Celeb-1M [15] dataset (10 million images of 100 thousand different identities) and fine-tuned on the VGGFace2 [11] dataset (3,31 million images of 9,131 different identities). The advantage of these network architectures is both in terms of efficiency and effectiveness. The ResNet models are four times faster than the VGGNet in performing the prediction of one image. Also, the size of the deep feature extracted from these networks (i.e. the output of the pool57x7_s1 layer) is 2048, that is half the size of the deep feature extracted from the VGGNet. As for the accuracy performances, we will study them in Section 5.

4 SYSTEM OVERVIEW

Our system is composed by two smart cameras, i.e. an integrated system composed by a processing board, a network interface (wired or wireless) and a video-camera, placed in two different offices in the CNR Area in Pisa. Each smart camera is placed inside the office in order to have a clear view of the entrance door and of the face of the entering person.

We used two Raspberry Pi 3 model B equipped with the Raspberry camera module v2 . The camera module has a Sony IMX219 8-megapixel sensor. The Raspberry Pi hardware includes:

- BCM2837 1.2GHz Quadcore ARM Cortex-A53, 64Bit CPU
- 1GB RAM DDR2
- 32GB micro SD card for storage

Figure 2: Pictures of one of the classes of the training set.

Figure 3: Samples of the ground truth dataset.

All the processing (image acquisition, face detection, feature extraction, and classification) is performed on board the smart camera. The only information transmitted is the output of the classification (authorized or not), that is sent to a server for visualization purposes.

4.1 Training Set and Ground Truth

In order to determine whether or not a person is allowed to enter into a specific area, we built both a training set and a ground truth. The training set is composed of ten pictures of the faces for each one of 44 different persons authorized to access the monitored area, for a total of 440 images. For each person, we used pictures that capture its face in different positions (face rotation of a few degrees with respect to the central axes, both horizontally and vertically) and with different facial expressions (see Figure 2).

Initially, we used ten pictures taken off-line before deploying the system where the person was in pose in front of the camera. However, we noticed that this approach did not work well because the off-line faces are very dissimilar to ones detected by the system at run-time. For example, the off-line pictures are very focused and high-resolution, while run-time pictures are often blurry and low-resolution since the faces are in the back of the scene. Also, the off-line pictures are all taken in a very short time interval, so they capture the person's face in the same physical and environmental conditions (beard length, glasses, hairstyle, brightness of the room, etc.). Therefore, we built another training set by using ten pictures for each person taken by the first set of images acquired on-line by the system (see Figure 2).

The ground truth is composed of 15,393 images of both authorized and unauthorized persons, captured by the two smart cameras placed in the offices in a test period of six months. Of these, $P = 14,380$ images belong to the 44 authorized persons, and $N = 1,013$ images belong to unknown/impostor people, which we know do not belong the group of 44 of the aforementioned identities. Figure 3 reports some examples of the ground truth. The faces reported in the figure have all been correctly classified with their true identity by our kNN classifier. Please note that both focused and blurred images have been correctly classified, as well as obstructed images and images with different beard conditions.

In both Figures 2 and 3, pictures have different sizes because we kept the original sizes of the faces as detected in the frames by the face detector.

4.2 Software description

The smart camera resolution is set to 1280x960 pixels, and a frame is captured every 2.9 seconds. The whole pipeline is implemented on board in Python with wrappers to OpenCV 3.4.1 (for SSD face detection) and to Caffe (for features extraction using the CNN). The detailed operations executed by our system are the following: in the first phase, face detection is executed to localize and crop the face from the captured image. We used the Single Shot Detector (SSD [22]) ResNet-10 model provided by OpenCV to perform face detection. For each detected face, we extract the deep feature by using OpenCV DNN module to load pre-trained Caffe models of both VGG-Face and VGGFace2 provided by the VGG group (http://www.robots.ox.ac.uk/ vgg/). We then compute the L2 distance between the query face deep feature and all the deep features in the training set, in order to perform the kNN classification. If the distance of the person returned by the kNN classifier is below a given threshold (we will show in the next Section how this threshold has been set), then the entering person is among the allowed ones and nothing happens. Otherwise, the system raises a notification of an unauthorized person entering a monitored office.

The code and other resources to replicate our setup are available here: https://www.github.com/fabiocarrara/deep-raspi-face.

5 EXPERIMENT EVALUATION

In order to validate the proposed system, we evaluated the performance of the intrusion detection system in terms of efficiency and effectiveness. Efficiency was measured evaluating the execution time in the various stages of the recognition process pipeline. Effectiveness was assessed by measuring false positive and false negative rate, and the classification accuracy.

5.1 Intrusion Detection Validation

The facial recognition task has been implemented using a k Nearest Neighbor (kNN) algorithm. We used each of the faces in our ground truth as a query for a kNN similarity search in the training set composed of ten representative images for each of the 44 authorized persons, and we extract the deep features from them using the three CNNs architectures described in Section 3. We used the Euclidean distance as a dissimilarity measure between features. We sorted the entire dataset of 440 features according to this distance with the given query from the nearest to the farthest, and we took the first k items. The result of this operation is the set K, of size k, of labeled faces belonging to the training set, ordered with respect to the increasing values of the distance. The label assigned to query by the classifier is the class that minimizes the sum of a similarity

Table 1: Average execution times on 10 runs of each stage of the pipeline. The ResNet-50 model is used for feature extraction.

Stage	mean ± std.dev.	details
Detection	0.64 ± 0.02	
PreProcess	0.11 ± 0.02	for each detected face
Extraction	2.11 ± 0.03	for each detected face
Matching	0.03 ± 0.00	for each detected face
Total	**2.89 ± 0.04**	(single face scenario)

Table 2: Confidence thresholds for FPR = FNR

Network Model	Threshold
VGGNet	0.254
SENet_ft	0.372
ResNet-50_ft	0.417

between the query and the faces labeled in the ranked list K [7]. The confidence of the classification is computed as $1 - d_f$, where d_f is the distance with the query of the first face of the winning class in K. The distances d_f are normalized so that they are always less than or equal to one, and $1 - d_f$ can be thought as the probability of the predicted label to be correct.

In order to find the optimal k, we computed the classification accuracy on the authorized faces of the ground truth (we called this subset P) for all the considered CNN models. We found that, in our scenario, the best k value for all the networks is 4.

5.1.1 Efficiency. A complete run that comprises capturing a new frame, performing the face detection, extracting the deep feature, and performing the kNN classification on the Raspberry Pi, takes approximately 2.9 seconds. Broken down times are reported in Table 1. From the table we can see that most of the execution time is spent in the feature extraction process, which is penalized by the fact that the GPU is not used. Even considering this, the overall response time is acceptable for the scenario we are considering.

5.1.2 False Positive and False Negative Rate analysis. We used a confidence threshold to decide whether a person is to be considered as authorized during the identification phase. If the value of probability for the most probable labels is below this threshold, the person's identity is claimed as unknown, and the classifier response is not authorized. In order to find the optimal confidence threshold, we evaluated the False Positive Rate $FPR = FP/N$ and the False Negative Rate $FNR = FN/P$, where FP is the number of False Positives, i.e., the number of unknown faces that have been wrongly authorized by the system, and FN is the number of False Negatives, i.e, the number of known persons wrongly not authorized by the system. N and P are, respectively, the subsets of not authorized and authorized persons of the ground truth.

When we deployed the system, we used as threshold the confidence value where the FPR is equal to FNR. Table 2 reports, for each network model studied, the confidence value that satisfies this condition.

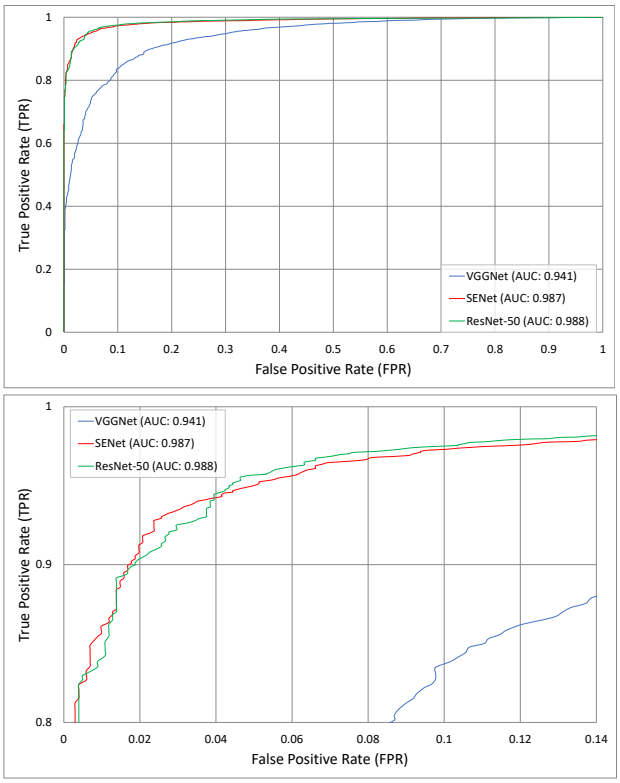

Figure 4: ROC curves for the three model considered. A higher value is better.

Figure 4 shows the Receiver Operating Characteristic (ROC) curves for all the models considered. The figure also reports their Area Under the Curve (AUC) values. The VGGNet is the model that achieved the worst result, with an AUC of 0.941. On the other hand, both SENet_ft and ResNet-50_ft work very well, with ResNet-50_ft slightly better than the SENet_ft with an AUC of 0.988, compared to the AUC of 0.987 of the SENet_ft.

We also computed the intrusion accuracy of the proposed systems for all the three network models considered. The intrusion accuracy is computed as $Acc = (TP + TN)/(P + N)$, where TP is the number of True Positives, i.e., the number of known faces that have been correctly authorized by the system, and TN is the number of True Negatives, i.e, the number of unknown persons correctly not authorized by the system. Table 3 reports the FNR (that is the error rate of the system in not allowing authorized people) and the intrusion accuracy of the three network models analyzed for fixed values of FPR of 1%, 5%, and 10%. Again, VGGNet shows the worst results in all cases, with FNR and intrusion accuracy rates much lower than the one obtained by the other models. SENet_ft and ResNet-50_ft are very close, and works quite good, obtaining a FNR of, respectively, 2.7% and 2.5% and an intrusion accuracy of 96.82% and 97.02%, in case of FPR = 0.1.

5.1.3 Classification accuracy. We also performed an evaluation of the accuracy rate of our approach in recalling the identity of the authorized people, the subset P of our ground truth (see Section

Table 3: Intrusion Performance Evaluation

Network Model	FPR=0.01		FPR=0.05		FPR=0.1	
	FNR	Acc	FNR	Acc	FNR	Acc
VGGNet	52.09%	51.28%	26.23%	75.17%	16.25%	84.16%
SENet_ft	**13.78%**	**87.07%**	5.05%	94.96%	2.7%	96.82%
ResNet-50_ft	15.85%	85.13%	**4.32%**	**95.64%**	**2.5%**	**97.02%**

Table 4: Classification Accuracy

Network Model	Accuracy
VGGNet	93.44%
SENet_ft	98.83%
ResNet-50_ft	**98.87%**

4.1). This is computed by ignoring the confidence threshold, and by counting, for each face in P, the number of correct predictions returned by the proposed classifier. Table 4 reports the accuracy value obtained by each of the network models analyzed. Also in this case, the best result is obtained by the ResNet-50_ft model which shows a classification accuracy of 98.87%, that is slightly better than the 98.83% accuracy rate of the SENet_ft model. Finally, VGGNet obtained a classification accuracy of 93.44%.

6 CONCLUSIONS

In this paper, we presented an approach to perform intrusion detection by using facial recognition on the off-the-shelf embedded device Raspberry Pi with the best (and free) available recognition technologies based on deep learning. In particular, we have used the 50-layers Residual Network (ResNet-50) trained on the VGGFace2 dataset, which has shown excellent accuracy performance without the need for GPUs support. Although the performance of the system in terms of computation times is acceptable, we plan in the future to use the Movidius Neural Compute Stick, which is an optimized miniature hardware and development platform for deep learning that should improve the processing speed of both face detection and features extraction.

7. REFERENCES

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. 2004. Face recognition with local binary patterns. *Computer vision-eccv 2004* (2004), 469–481.

[2] T. Ahonen, A. Hadid, and M. Pietikainen. 2006. Face description with local binary patterns: Application to face recognition. *IEEE transactions on pattern analysis and machine intelligence* 28, 12 (2006), 2037–2041.

[3] H. Ai and T. Li. 2017. A smart home system based on embedded technology and face recognition technology. *Intelligent Automation & Soft Computing* 23, 3 (2017), 405–418.

[4] C. Alippi, S. Disabato, and M. Roveri. 2018. Moving convolutional neural networks to embedded systems: the alexnet and VGG-16 case. In *17th ACM/IEEE Intnl Conference on Information Processing in Sensor Networks*. IEEE Press, 212–223.

[5] G. Amato, F. Carrara, F. Falchi, C. Gennaro, C. Meghini, and C. Vairo. 2017. Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications* 72 (2017), 327–334.

[6] G. Amato, F. Carrara, F. Falchi, C. Gennaro, and C. Vairo. 2016. Car parking occupancy detection using smart camera networks and deep learning. In *Computers and Communication (ISCC), 2016 IEEE Symposium on*. IEEE, 1212–1217.

[7] G. Amato, F. Falchi, and C. Gennaro. 2015. Fast image classification for monument recognition. *Journal on Computing and Cultural Heritage* 8, 4 (2015), 18.

[8] G. Amato, F. Falchi, C. Gennaro, and C. Vairo. 2018. A Comparison of Face Verification with Facial Landmarks and Deep Features. In *Proceedings of the 10th International Conference on Advances in Multimedia (MMEDIA 2018)*. 1–6.

[9] G. Amato, F. Falchi, and L. Vadicamo. 2016. Visual recognition of ancient inscriptions using convolutional neural network and fisher vector. *Journal on Computing and Cultural Heritage (JOCCH)* 9, 4 (2016), 21.

[10] Y. Bengio. 2011. Deep Learning of Representations for Unsupervised and Transfer Learning. In *Proceedings of the 2011 International Conference on Unsupervised and Transfer Learning Workshop - Volume 27 (UTLW'11)*. JMLR.org, 17–37. http://dl.acm.org/citation.cfm?id=3045796.3045800

[11] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. 2017. VGGFace2: A dataset for recognising faces across pose and age. *arXiv:1710.08092* (2017).

[12] G. E. Dahl, D. Yu, L. Deng, and A. Acero. 2012. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing* 20, 1 (2012), 30–42.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik. 2016. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence* 38, 1 (2016), 142–158.

[14] T. S. Gunawan, M. H. H. Gani, F. D. A. Rahman, and M. Kartiwi. 2017. Development of Face Recognition on Raspberry Pi for Security Enhancement of Smart Home System. *Indonesian Journal of Electrical Engineering and Informatics (IJEEI)* 5, 4 (2017), 317–325.

[15] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. 2016. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *European Conference on Computer Vision*. Springer, 87–102.

[16] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[17] J. Hu, L. Shen, and G. Sun. 2017. Squeeze-and-excitation networks. *arXiv:1709.01507* (2017).

[18] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. 2007. *Labeled faces in the wild: A database for studying face recognition in unconstrained environments*. Technical Report. Technical Report 07-49, University of Massachusetts, Amherst.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[20] Y. Lecun, Y. Bengio, and G. Hinton. 2015. Deep learning. *Nature* 521, 7553 (5 2015), 436–444. DOI: http://dx.doi.org/10.1038/nature14539

[21] J. Liu, Y. Deng, T. Bai, Z. Wei, and C. Huang. 2015. Targeting ultimate accuracy: Face recognition via deep embedding. *arXiv:1506.07310* (2015).

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*. Springer, 21–37.

[23] O. M. Parkhi, A. Vedaldi, and A. Zisserman. 2015. Deep Face Recognition. In *British Machine Vision Conference*.

[24] M. Sahani, C. Nanda, A. K. Sahu, and B. Pattnaik. 2015. Web-based online embedded door access control and home security system based on face recognition. In *Circuit, Power and Computing Technologies (ICCPCT), 2015 International Conference on*. IEEE, 1–6.

[25] A. V. Savchenko and N. S. Belova. 2018. Unconstrained Face Identification Using Maximum Likelihood of Distances Between Deep Off-the-shelf Features. *Expert Systems with Applications* (2018).

[26] F. Schroff, D. Kalenichenko, and J. Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 815–823.

[27] K. Simonyan and A. Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556* (2014).

[28] L. Vadicamo, F. Carrara, A. Cimino, S. Cresci, F. Dell'Orletta, F. Falchi, and M. Tesconi. 2017. Cross-Media Learning for Image Sentiment Analysis in the Wild. In *2017 IEEE Intnl. Conference on Computer Vision Workshops (ICCVW)*. 308–317.

[29] B. Vaidya, A. Patel, A. Panchal, R. Mehta, K. Mehta, and P. Vaghasiya. 2017. Smart home automation with a unique door monitoring system for old age people using Python, OpenCV, Android and Raspberry pi. In *Intelligent Computing and Control Systems (ICICCS), 2017 International Conference on*. IEEE, 82–86.

[30] Y. Wang, C. von der Weth, T. Winkler, and M. Kankanhalli. 2016. Tweeting Camera: A New Paradigm of Event-based Smart Sensing Device. In *Proceedings of the 10th International Conference on Distributed Smart Camera*. ACM, 210–211.