Empirical Comparison of Face Verification Algorithms from UAVs

Julio Diez-Tomillo
University of the West of Scotland
Paisley, United Kingdom
0000-0003-1979-3592

Jose M. Alcaraz-Calero
University of the West of Scotland
Paisley, United Kingdom
0000-0002-2654-7595

Qi Wang
University of the West of Scotland
Paisley, United Kingdom
0000-0002-7764-9858

***Abstract*—Face verification use cases have recently gained momentum in the increasingly digitalised society, and thus the need arises significantly to integrate this technology in wireless/mobile networked systems such as 5G and applications such as Unmanned Aerial Vehicle (UAV) based public safety services. However, there is no benchmarking result for the evaluation of the various existing face verification algorithms for such UAV applications. This paper is concerned with such new use cases (e.g., the Drone Guard Angel in the EU H2020 project ARCADIAN-IoT and the surveillance network applications in the EU H2020 project 5G-INDUCE), and provides an empirical comparison among three popular state-of-the-art face verification algorithms for this use case. To this end, a face verification pipeline is presented. These algorithms are then compared in terms of their inference time, and the distribution of the similarity indexes for different distances in UAV-based use cases. Their strengths and weaknesses are analysed, leading to an insightful recommendation on their applicability scenarios for UAVs.**

***Index Terms*—Face verification, UAV, Similarity index, Cosine distance, Inference speed**

I. INTRODUCTION

Nowadays, new technologies are emerging due to the digitisation of society. Two of them are face recognition and face verification. The difference between them is that face recognition needs a database because it relies on telling which person a face belongs to. On the other hand, face verification just compares two faces and decides whether the face belongs to the same person or to a different one.

In recent years face verification has become crucial in various applications, including security and surveillance systems or biometric identification. Traditional face verification systems are based on static cameras in fixed locations. But they have several limitations like restricted coverage or the inability to track a person. In contrast, the use of UAVs for face verification can solve those limitations by being able to cover large areas and follow a person in real-time. One main new use case using this technique is the Drone Guard Angel for public safety escort services, as being developed in the EU H2020 project ARCADIAN-IoT [1]. The UAV will go to the position of the user requesting the service, it will verify the face of the person and will accompany the person home ensuring that it arrives safely. Another example can be observed in the surveillance network applications in the EU H2020 project 5G-INDUCE [2], where the pilot of the UAV can be verified before the authorised operation of the UAV through this method.

These use cases arose the need for face verification algorithms that can perform accurately with video feeds from UAVs. This means at far distances from the person and therefore low-pixel faces. Moreover, the UAVs would have to verify in environmental challenges conditions such as poor lighting for outdoor and late evenings operations and also fast-changing poses and positions to the target. Unfortunately, there is a serious gap in the literature on face verification at far distances from UAVs. Thus, this paper analyses different state-of-the-art face verification algorithms and compares the three best ones. For this purpose, their accuracy and inference time is going to be compared using a dataset with UAV-recorded videos.

The main contribution of this research is an empirical comparison of three state-of-the-art face verification algorithms for a UAV-based use case. A dataset containing videos recorded from a UAV has been created for this comparison. The videos have been recorded from four different distances to analyse how these algorithms behave as the person to be verified is further away. The inference time of the face verification algorithm is also analysed to be able to do a precise accuracy/inference time comparison. This will allow us to analyse whether the face verification algorithms are optimal for UAV-based use cases. In summary, the main contributions of this study are as follows:

- Design and implementation of a face verification pipeline to conduct the experiments in similar conditions.
- Creation of a UAV recorded dataset for comparison of three state-of-the-art face verification algorithms.
- An empirical analysis of the inference time and similarity indexes obtained for four different distances using three state-of-the-art face verification algorithms.
- A comparison of the inference time in a face verification pipeline using three state-of-the-art algorithms.
- An empirical comparison among three state-of-the-art face verification algorithms, leading to practical recommendations of their applicability in UAV-based use cases.

The remainder of this paper is as follows: section II reviews the state-of-the-art face verification algorithms and explains why those three have been selected. Section III shows the experimental setup to carry out the experiments. Experiments and results are shown and discussed in section IV. Concluding remarks are given in section V.

II. Related Work

UAVs are a technology whose use is increasing exponentially. They are making it easier to perform previously difficult tasks. For instance, now UAVs can detect lost people in the forest from the sky whereas before it had to be done by helicopter with the cost that this entailed [16] [17].

Some papers explore the use of UAVs to perform face verification tasks. They usually use simple algorithms that do not have great accuracy but they are fast, such as LBPH (Local binary patterns histograms) [18] which was developed in 1996. Most of these algorithms are used because of their small capabilities needed to execute them as the system is embedded in the UAV. However, suppose a high-speed connection is made between a ground control station (GCS) and the UAV. In that case, high-performance algorithms can be used as they will be executed in the GCS and the UAV will only transmit the video. The UAV can be connected to the GCS, for instance, via a 5G connection [19] which also allows controlling the UAV with BVLOS (Beyond Visual Line of Sight) flights.

Moreover, previous studies have explored the impact of distance and height on the accuracy of face recognition algorithms using an UAV [20]. Another line of research has performed face recognition and distance estimation from UAVs using a siamese network with a ResNet v1 architecture [21]. Furthermore, the architectures ResNet-50 and SENet have been compared for face recognition on UAVs [22].

Table I shows a comparison between different state-of-the-art face verification algorithms. Different parameters are compared such as the input and output size of the neural network, and the number of images used for training. Also, the verification performance is going to be compared using two different datasets: Labeled Faces in the Wild (LFW) [3] and Youtube Faces (YTF) [4].

The first row of the table shows the human performance on the LFW dataset which is a 97.53% [5]. This value is going to be used to have a first reference of how good an algorithm is. Only two algorithms of the table do not surpass the verification performance of human beings on the LFW dataset: DeepFace [6] achieving a close 97.35% and OpenFace [7] that is far with only a 92.92%. The rest of the algorithms surpass the human-beings verification performance.

The ones that perform the best in this dataset are CosFace [9] (99.73%), DeepID2 [10] (99.53%), ArcFace [11] (99.83%) and FaceNet512 [13] (99.60%), all of them achieving more than 99.5%. On the other hand, not all algorithms have been evaluated using the YTF dataset, therefore not all the values are on the table. ArcFace (98.02%), CosFace (97.60%) and VGG-Face [8] (97.30%) have the best verification performance on this dataset.

Both FaceNet algorithms have been trained with the biggest number of images (200 million). The difference is that FaceNet512 [13] is an extended version of FaceNet [12] that has a 512 vector as output instead of 128, increasing the verification performance as can be seen in the table.

For the purpose of this research, three algorithms of table I have been selected to perform an analysis on UAV-based use cases: ArcFace [11], VGG-Face [8] and FaceNet512 [13]. They have been chosen due to the good verification performance they have on the LFW and the YTF dataset. Also, they have all surpassed the human-being verification performance on LFW.

Furthermore, many metrics can be used to calculate the similarity indexes between two faces, such as euclidean distance, Manhattan distance, and cosine distance. This paper is going to use the last one. It is one of the most used in the literature as it is able to show the similarity of two vectors more accurately. Cosine distance is calculated as follows:

$$CD(\vec{x},\vec{y}) = 1 - \frac{\sum_1^n x_i y_i}{\sqrt{\sum_1^n x_i^2}\sqrt{\sum_1^n y_i^2}}$$

Where x and y are the features vector of two different faces. And the result of the calculation is the similarity index between them.

III. Experimental Setup

A. Dataset

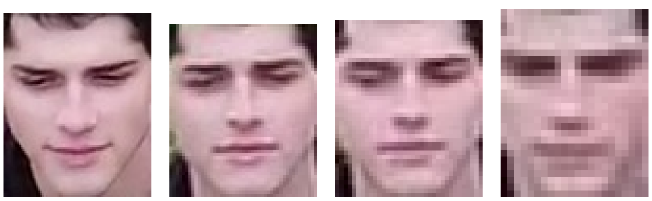
There is a lack of datasets that meet the requirements of our case (Drone Guard Angel), which focuses on verifying a person standing and looking at the UAV. In the literature, there is a limited number of datasets available for face verification with UAVs. However, these datasets are not suitable for our research for different factors such as lack of different ethnicities, videos not recorded using UAVs, or a limited number of faces. For instance, the DroneSurf dataset [23], while comprehensive and useful for face recognition in surveillance applications, does not meet the specific requirements of our use case. Moreover, our use case needs close images of each person to be verified as that picture needs to be compared with the ones from the UAV to perform the face verification. This dataset will not be released due to data protection constraints, as the authorisation of the volunteers was only given for this conference study.

Therefore, a new dataset of UAV-recorded humans has been created for this conference. It was created using a DJI Mini 2 drone and includes videos of 20 volunteer individuals of different ethnicities. The volunteers have been recorded at four different distances: 5, 7, 10, and 15 meters. These distances were selected first to be at a safe distance from the volunteer to record the videos. And also, to not be excessively far to not be able to detect the face using the RetinaFace [24] face detection algorithm. The recording angle is 30 degrees so that they did not have to raise their head excessively and maintain an optimal height for the UAV. Each video recorded lasts 30 seconds, during which the volunteer was asked to make different head movements to acquire a comprehensive range of data on different facial angles. The videos have a resolution of 4K (3840x2160) and are captured at 30 fps. Moreover, the dataset also contains a close image of each volunteer to be able to compare it with the faces from the videos to perform

TABLE I
COMPARATIVE BETWEEN DIFFERENT FACE VERIFICATION STATE-OF-THE-ART ALGORITHMS

Ref	Algorithm	Input Size	Output Size	Verification Accuracy		Training Images
				on LFW dataset [3]	on YTF dataset [4]	
[5]	***Human-beings***	NA	NA	97.53%	Not given	NA
[6]	**Deep-Face**	152x152x3	4096	97.35%	91.40%	4.4M
[7]	**OpenFace**	96x96x3	128	92.92%	Not given	0.5M
[8]	**VGG-Face**	224x224x3	2622	98.95%	97.30%	2.6M
[9]	**CosFace**	112x96x3	Not given	99.73%	97.60%	5M
[10]	**DeepID2**	55x47x3	160	99.53%	93.20%	0.2M
[11]	**ArcFace**	112x112x3	512	99.83%	98.02%	5.8M
[12]	**FaceNet**	160x160x3	128	98.87%	95.12%	200M
[13]	**FaceNet512**	160x160x3	512	99.60%	Not given	200M
[14]	**Dlib**	150x150x3	128	99.38%	Not given	3M
[15]	**SphereFace**	112x96x3	512	99.42%	95.00%	0.5M

the face verification. Fig. 1 shows one example of a cropped face from our dataset and each of the distances - 5, 7, 10 and 15 meters.



(a) 5 m (b) 7 m (c) 10 m (d) 15 m

Fig. 1. Example of a cropped face at the four distances recorded in the dataset

B. Design of the pipeline

One of the main contributions is the design and implementation of a pipeline presented in Fig. 2. As input, it has two images. The first one is a frame of a video recorded from a UAV obtained from our dataset videos. The second input is the face of the person we want to verify. This image has been obtained from a phone at a close distance. The output of our pipeline is the similarity index between the face of the person to verify and the faces in the video frame. It is divided into four stages as follows:

1) **Face Detection:** This is the first stage of our pipeline. The face detection algorithm used is RetinaFace as it has the best accuracy while detecting at long distances. On the other hand, it is not a fast algorithm compared to other face detection ones. But as the purpose of this study is to compare only the face verification algorithms this is not a problem. RetinaFace receives as input an image and provides the coordinates of every face in it and the accuracy of the detection.
2) **Preprocessing:** It is divided into two steps. The first one crops the face from the image using the coordinates provided by RetinaFace. Then the face is resized to the input size required by the face verification algorithm. The input size varies depending on the algorithm as shown in Table I. As the ratio of the input size is not going to be usually the same as the face, it is needed to apply padding to that image. Therefore, black pixels are included to reach the expected input size maintaining the image ratio. This ensures that the face is not distorted.
3) **Siamese Network [6] [25]:** They have been used by several authors in the literature to perform face verifications. A siamese network is composed of two Convolutional Neural Networks (CNN) that have the same architecture and weights. If the input of both CNNs is the same, the output will also be the same. The input of each CNN is a face sized as the input size requested. The output will be the features of that face. These features are a vector of different dimensions depending on the algorithm used that represents the characteristics of that face.
4) **Similarity index calculation:** It is the last stage of the pipeline. It receives the features of both faces from the siamese network and calculates its similarity index. It is obtained by calculating the cosine distance between them as seen in the previous section. Thus, if we obtain a similarity index of 0 means that the faces are exactly alike while if the result is 2 means that the two faces are opposite.

C. Implementation

The results have been obtained using a common framework where all three face verification algorithms are implemented. The implemented framework is referred to as DeepFace [26] [27] and includes the three face verification algorithms, RetinaFace for face detection and cosine distance to calculate the similarity indexes. Evaluating all the algorithms on the same platform allows us the obtain reliable results to be compared as they have been obtained under the same conditions.

DeepFace has been run on Python version 3.8. It is powered mainly by Keras [28]. Also, the Python library OpenCV has been used for the preprocessing and postprocessing of the images. It has been executed on a computer with Focal Ubuntu version 20.04.3.

The pipeline has been implemented and executed using the three state-of-the-art face verification algorithms on the same testbed for evaluation and comparison purposes using the mentioned framework. The algorithms are going to be executed in a high-performance GPU for comparison. Thus, it

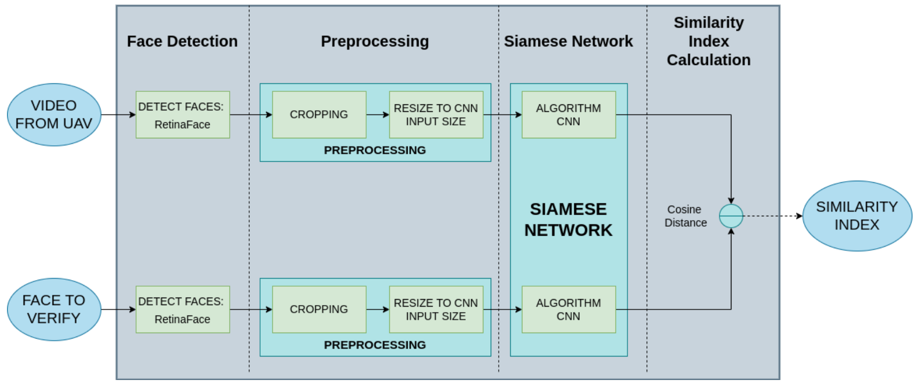


Fig. 2. Pipeline used to obtain the results from the experiments. It is divided into four stages: face detection, preprocessing, siamese network, and similarity index calculation.

can be simulated that the video is sent from the UAV to a high-performance GPU server for the execution of the pipeline.

IV. Experiments and Results

A. Testbed Description

The experiments have been carried out on a NVIDIA GeForce GTX TITAN X with 12GB of onboard memory. These experiments have been repeated 10 times to obtain the final results. The UAV used to record the dataset is a DJI Mini 2 that has a 4K camera (3840x2160 px) at 30 fps.

B. Comparative

The three face verification algorithms will be compared using two metrics - inference time, and the similarity index distribution for positive and negative pairs - at four different fixed distances.

1) Inference time: It is defined as the time it takes for one frame to complete a full pipeline execution. The inference time can be divided into four stages, the same as the processing pipeline. This is useful to compare only the face verification algorithms' inference time and not the whole pipeline.

TABLE II
Pipeline inference time comparison of each face verification algorithm

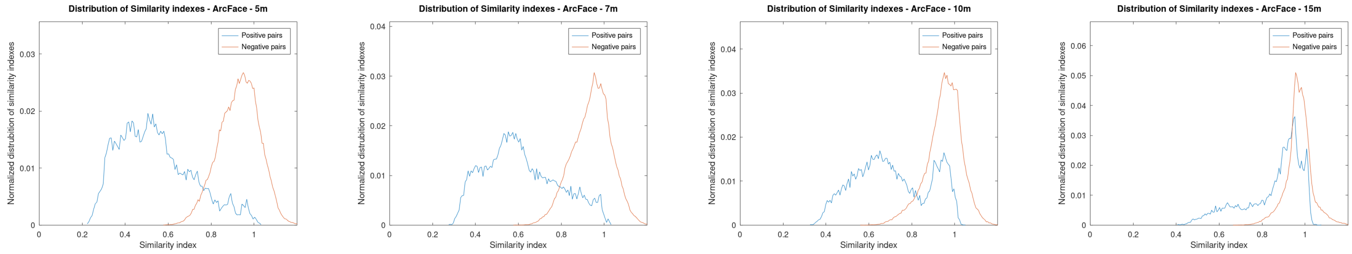
Algorithm	ArcFace	FaceNet512	VGG-Face
Times	230 ms	240 ms	280 ms

1) **Face Detection**: Using the RetinaFace algorithm it takes around 165 ms to perform the face detection. Using another faster face detection algorithm, the inference time of the pipeline can be reduced greatly. This stage is the slowest in the pipeline.
2) **Preprocessing**: The time is independent of the face detection or verification algorithms used. This stage only takes 0.3 ms to run. It is the fastest of the four stages.
3) **Siamese Network**: Only the time it takes to obtain the features of the face from the video is taken into account. This is because the features of the image of the person to be verified are obtained at the beginning of the video and are not recalculated to accelerate the process. As the image is the same throughout the video, the same results are obtained as if they have been calculated for each frame. The time this stage takes depends on the algorithm used, as Table II reflects.
4) **Distance calculation**: The last stage is also independent of the algorithms used. Using cosine distance as a metric it takes around 1.4 ms.

All stages have fixed times independent of the face verification algorithm used except for the Siamese network. The time differences that can be seen in Table II are exclusively due to which face verification algorithm is used. VGG-Face is by far the slowest face verification algorithm and ArcFace the fastest. All stages excluding the Siamese network take approximately 166.7 milliseconds. Therefore, the inference times of only the siamese network for the three face verification algorithms are VGG-Face 113.3 ms, FaceNet512 73.3 ms, and ArcFace 63.3 ms.

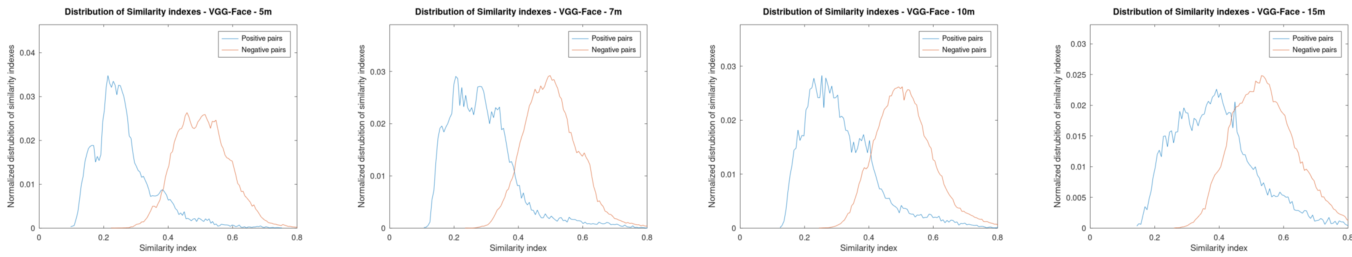
2) Similarity indexes distribution: The pipeline in the previous section has been used to obtain the similarity indexes of our dataset. They have been divided into two types: positive and negative pairs. The first ones are obtained by calculating the similarity index of two faces that belongs to the same person. The negative ones have been obtained using two faces that belong to two different people, so the verification will be negative. The plots of both similarity indexes are shown in each graphic. The further apart the two plots are, the better the algorithm is because it will be able to perform verifications with fewer false positives and negatives. If the plots are overlapped, it will be difficult to differentiate between the positive and negative pairs so there will be a lot of false positives and negatives.

The plots have been obtained for each of the three face verification algorithms: ArcFace (Figure 3), VGG-Face (Figure 4) and FaceNet512 (Figure 5). Four different distances (5, 7, 10, and 15 meters) have been used to see how the plots evolve depending on the distance. By looking at all the graphs it can be appreciated that as the distance increases, the plots shift to the right. The similarity indexes are higher because at a greater distance, the face images have less resolution, so they



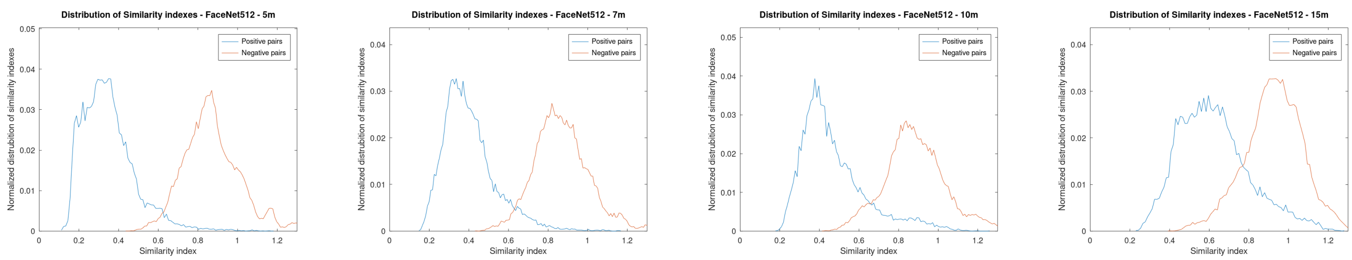
(a) 5 meters (b) 7 meters (c) 10 meters (d) 15 meters

Fig. 3. Similarity indexes of the ArcFace face verification algorithm for 5, 7, 10 and 15 meters of distance



(a) 5 meters (b) 7 meters (c) 10 meters (d) 15 meters

Fig. 4. Similarity indexes of the VGG-Face face verification algorithm for 5, 7, 10 and 15 meters of distance



(a) 5 meters (b) 7 meters (c) 10 meters (d) 15 meters

Fig. 5. Similarity indexes of the FaceNet512 face verification algorithm for 5, 7, 10, and 15 meters of distance

become less similar.

Let us focus first on the ArcFace graphs (Figure 3). The 5 meters graph (Figure 3a) shows both plots - positive and negative pairs - well separated. The negative pairs curve starts approximately at 0.6 while most positive pair similarity indexes are below this point. So at this distance, users can be verified with high accuracy. At 7 meters (Figure 3b) and 10 meters (Figure 3c) the plots are similar but shifted to the right. The negative pairs curve continues starting at 0.6, but the positive pairs have higher similarity indexes. Most of the similarity indexes are above 0.6, so it will be more difficult to correctly verify a person without false positives. Furthermore, the 15 meters graph (Figure 3d) shows how both curves are almost completely overlapped. Thus, it will be difficult to perform face verifications correctly. Only a small part of the curve is not overlapped below 0.7. So if the similarity index is below this value we will be able to perform face verifications without any false positives, therefore, it can be a good value to establish the verification threshold.

VGG-Face (Figure 4) obtains good results in graphs at 5 (Figure 4a) and 7 meters (Figure 4b) as the negative and positive pairs can be difference easily. At 10 meters (Figure 4c) the curves are more overlapped but a significant part of the positive pair curve is still below the minimum negative pair similarity index. Therefore, most of the face verifications will be performed without any false positives by choosing an appropriate threshold. At 15 meters (Figure 4d) the curves overlap a lot, so there will be many false positives and negatives regardless of the chosen threshold.

FaceNet512 is the one that achieves the best results (Figure 5). The graphs at all distances are well separated so it can be easier to verify a person than the other algorithms. At 5 meters (Figure 5a) most of the positive pair curve is below 0.5, which is the minimum negative pair similarity index, so almost all of the verifications will be performed correctly without false positives. Furthermore, at 7 (Figure 5b) and 10 meters (Figure 5c) the peak of the positive pairs is still below the minimum negative pair similarity index. At 15 meters (Figure 5d) the curves are more overlapped but they can still be differenced easily. Thus, there will be false positives but not as many as using the other face verification algorithms.

V. Concluding remarks

This study has presented a comparison using a novel created dataset designed for this concrete purpose. Three state-of-the-art face verification algorithms have been compared at four different distances (5, 7, 10, and 15 meters). Two metrics have been compared: the inference time processing a frame in the whole pipeline, and the distribution of their similarity indexes for both positive and negative pairs.

Based on this comparison, it is possible to conclude if the state-of-the-art face verification algorithms are optimal to perform face verifications in real-time from UAVs. ArcFace is a fast algorithm but it only achieves good results up to 10 meters of distance. On the other hand, FaceNet512 can verify correctly at all analyzed distances but it is slower than the previous one. Finally, VGG-Face is a slow algorithm so it is only recommended to be used when the speed is not important in the use case. Further work would involve choosing one of the previously compared algorithms to perform face verification on UAVs. The results and video will be transmitted to the user via a wireless network such as 5G.

Acknowledgment

This work was in part funded by the EU Horizon 2020 5G-PPP 5G-INDUCE project ("Open cooperative 5G experimentation platforms for the industrial sector NetApps") under the Grant number H2020-ICT-2020-2/101016941, the EU Horizon 2020 ARCADIAN-IoT project ("Autonomous Trust, Security and Privacy Management Framework for IoT") under the Grant number H2020-SU-DS-2020/101020259, and the EU Horizon INCODE project ("Programming Platform for Intelligent Collaborative Deployments over Heterogeneous Edge-IoT Environments") under the Grant number HORIZON-CL4-2022-DATA-01-03/101093069. The authors would like to thank all the partners in these projects for their support.

References

[1] E. H. project ARCADIAN-IoT, "Autonomous trust, security and privacy management framework for iot."

[2] E. H. project 5G-INDUCE, "Open cooperative 5g experimentation platforms for the industrial sector netapps."

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

[4] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *CVPR 2011*, 2011, pp. 529–534.

[5] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 365–372.

[6] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[7] B. Amos, B. Ludwiczuk, and M. Satyanarayanan, "Openface: A general-purpose face recognition library with mobile applications," CMU-CS-16-118, CMU School of Computer Science, Tech. Rep., 2016.

[8] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proceedings of the British Machine Vision Conference (BMVC)*, M. W. J. Xianghua Xie and G. K. L. Tam, Eds. BMVA Press, September 2015, pp. 41.1–41.12. [Online]. Available: https://dx.doi.org/10.5244/C.29.41

[9] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, "Cosface: Large margin cosine loss for deep face recognition," 2018. [Online]. Available: https://arxiv.org/abs/1801.09414

[10] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," 2014.

[11] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," 2019.

[12] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[13] D. Sandberg, "Face recognition using tensorflow," 2018. [Online]. Available: https://github.com/davidsandberg/facenet

[14] D. E. King, "Dlib-ml: A machine learning toolkit," *Journal of Machine Learning Research*, vol. 10, no. 60, pp. 1755–1758, 2009. [Online]. Available: http://jmlr.org/papers/v10/king09a.html

[15] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," 2017. [Online]. Available: https://arxiv.org/abs/1704.08063

[16] I. Martinez-Alpiste, G. Golcarenarenji, Q. Wang, and J. M. Alcaraz-Calero, "Search and rescue operation using uavs: A case study," *Expert Systems with Applications*, vol. 178, p. 114937, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741742100378X

[17] G. Golcarenarenji, I. Martinez-Alpiste, Q. Wang, and J. M. Alcaraz-Calero, "Efficient real-time human detection using unmanned aerial vehicles optical imagery," *International Journal of Remote Sensing*, vol. 42, no. 7, pp. 2440–2462, 2021. [Online]. Available: https://doi.org/10.1080/01431161.2020.1862435

[18] A. Saha, A. Kumar, and A. K. Sahu, "Face recognition drone," in *2018 3rd International Conference for Convergence in Technology (I2CT)*, 2018, pp. 1–5.

[19] D. Mishra and E. Natalizio, "A survey on cellular-connected uavs: Design challenges, enabling 5g/b5g innovations, and experimental advancements," *Computer Networks*, vol. 182, p. 107451, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128620311324

[20] M. Rostami, A. Farajollahi, and H. Parvin, "Deep learning-based face detection and recognition on drones," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–15, 2022.

[21] Y.-H. Pu, P.-S. Chiu, Y.-S. Tsai, M.-T. Liu, Y.-Z. Hsieh, and S.-S. Lin, "Aerial face recognition and absolute distance estimation using drone and deep learning," *The Journal of Supercomputing*, vol. 78, no. 4, pp. 5285–5305, 2022.

[22] G. Amato, F. Falchi, C. Gennaro, F. V. Massoli, and C. Vairo, "Multi-resolution face recognition with drones," in *2020 3rd International Conference on Sensors, Signal and Image Processing*, 2020, pp. 13–18.

[23] I. Kalra, M. Singh, S. Nagpal, R. Singh, M. Vatsa, and P. B. Sujit, "Dronesurf: Benchmark dataset for drone-based face recognition," in *2019 14th IEEE International Conference on Automatic Face Gesture Recognition (FG 2019)*, 2019, pp. 1–7.

[24] J. Deng, J. Guo, Y. Zhou, J. Yu, I. Kotsia, and S. Zafeiriou, "Retinaface: Single-stage dense face localisation in the wild," 2019.

[25] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 539–546.

[26] S. I. Serengil and A. Ozpinar, "Lightface: A hybrid deep face recognition framework," in *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*. IEEE, 2020, pp. 23–27. [Online]. Available: https://doi.org/10.1109/ASYU50717.2020.9259802

[27] ——, "Hyperextended lightface: A facial attribute analysis framework," in *2021 International Conference on Engineering and Emerging Technologies (ICEET)*. IEEE, 2021, pp. 1–4. [Online]. Available: https://doi.org/10.1109/ICEET53442.2021.9659697

[28] F. Chollet *et al.*, "Keras," 2015. [Online]. Available: https://github.com/fchollet/keras