ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

DEPARTMENT OF COMPUTER SCIENCE
AND ENGINEERING

ARTIFICIAL INTELLIGENCE

MASTER THESIS

in

Image Processing and Computer Vision

SMART MIRROR: REMOTE VERIFICATION OF AN IDENTITY DOCUMENT HOLDER

CANDIDATE
Michele Vece

SUPERVISOR
Prof. Luigi Di Stefano

Academic Year 2021-2022

Session 3rd

Contents

List of Figures

Abstract

Nowadays, some activities, such as subscribing an insurance policy or opening a bank account, are possible by navigating through a web page or a downloadable application.

Since the user is often “hidden” behind a monitor or a smartphone, it is necessary a solution able to guarantee about their identity.

Companies are often requiring the submission of a “proof-of-identity”, which usually consists in a picture of an identity document of the user, together with a picture or a brief video of themselves.

This work describes a system whose purpose is the automation of these kinds of verifications.

Chapter 1

Introduction

This work was carried out in collaboration with Blue Reply srl [1]. It consists in the implementation of a face detection and verification web application, which should integrate their already working document recognition app.

1.1 Content overview

This work is structured as follows.

Chapter 2 explains the concepts of face detection and face verification and briefly describes the main characteristics of the architectures used in next sections.

Chapter 3 describes the implementation of the web application and its architecture and illustrates a simple use case.

Chapter 4 goes deep in the choice of some parameters of the system and contains the analysis of its behaviour in response to freely available datasets and the evaluation on a custom dataset.

Chapter 5 resumes the entire work and presents the conclusions drawn.

Chapter 2

Literature review

2.1 Face Recognition

The term Face Recognition (FR) refers to systems capable of detecting and matching a human face from a digital image or a video frame against a set of faces.

2.1.1 Face Recognition before neural networks

Face Recognition approaches prior to the spread of neural networks could be divided into three main groups [12].

First, holistic approaches, which use a derivation of low-dimensional representation through certain distribution assumptions. Second, local-feature-based FR, that make use of some invariant properties of local filtering. Finally, learning-based local descriptors, in which local filters are learnt for better distinctiveness.

Differently from neural networks, these traditional methods attempted to recognize human faces by few layers of representations. In addition, most methods aimed to address only one aspect of unconstrained facial changes: there was no method to address these unconstrained challenges in an integrated way. As a consequence, these "shallow" methods were not capable to

extract stable identity features invariant to real-world variations.

2.1.2 Face Recogniton neural networks

Face Recognition networks inherit from deep classification networks [12]: the network is trained over a set of known face identities and then an intermediate bottleneck layer is used as a representation to generalize recognition beyond the set of identities used in the training.

Evolution of network architectures and training losses

As a consequence, the first architectures were exclusively inspired by standard Convolutional Neural Networks. However, in the years, they have been evolving and nowadays include also inception modules or residual layers.

Similarly, initially FR networks adopted cross-entropy based softmax loss for feature learning. However, this loss was not sufficient by itself to learn discriminative features. An alternative approach consisted in the use of Euclidean-distance-based losses (e.g., contrastive loss, triplet loss). Starting from 2017, angular and cosine-margin-based losses became popular.

2.1.3 Face Recognition challenges

Many challenges are still open in the face recognition paradigm. They include dealing with different poses, ages, make-up and low resolution images.

2.2 Face Detection

The term Face Detection refers to systems capable of detecting a human face in a digital image or a video frame.

They usually return the coordinates of the detected face(s), if any, and a confidence value.

2.2.1 Adopted model

In the following, a brief overview of the model adopted as face detector in the web application.

MTCNN

Multitask Cascaded Convolutional Neural Networks (MTCNN) are described in [13].

The cascade face detector proposed by Viola and Jones [11] and based on Haar-Like features usually achieves good performance with real-time efficiency. However, it may degrade significantly in real-world applications with larger visual variations of human faces. As a consequence, a more robust detector is needed and MTCNN is a good alternative.

In MTCNN, given an image, it is resized to different scales to build an image pyramid, which is the input of the following three-stage cascaded framework:

1. a Proposal Network (P-Net), which identifies the candidate windows and their bounding box regression vectors;
2. a Refine Network (R-Net), which further rejects a large number of false candidates;
3. an Output Network (O-Net), which outputs the five facial landmarks' positions for eyes, nose and mouth.

At each step, calibration with bounding box regression is performed and highly overlapped candidates are merged via non-maximum suppression.

2.3 Face Verification

Face Verification aims at establishing whether two images represent the same person.

It differs from Face Identification, whose purpose, instead, consists in determining the specific identity of the person represented in the image. While Face Verification computes *one-to-one* similarity between the two input images, Face Identitification performs *one-to-many* similarity.

2.3.1 Adopted models

In the following, a brief overview of the models used in the web application.

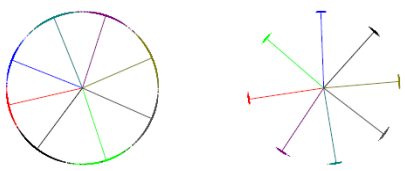
ArcFace

ArcFace is described in [3]. The architecture implemented in [10] is a ResNet-34 with an embedding dimension of 512 and reaches 99.41% on LFW benchmark.

They propose an Additive Angular Margin Loss to further improve the discriminative power of the face recognition model and the stabilization of the training process.

The arc-cosine function is used to calculate the angle between the current feature and the target weight, then an additive angular margin penalty is added to simultaneously enhance the intra-class compactness and inter-class discrepancy.

As a consequence, while softmax loss provides roughly separable feature embedding, but produces ambiguity in decision boundaries, ArcFace loss enforces a more evident gap between the nearest classes, as shown in figure 2.1 from [3].



(a) Softmax (b) ArcFace

Figure 2.1: ArcFace leads to better separation between closest classes

Dlib

Dlib face recognition network is briefly described in [5]. Also its architecture is a ResNet, but with 29 layers convolutional layers. The dimension of the embedding is 128 and it reaches 99.38% on LFW benchmark.

According to its creator, it is essentially a version of the ResNet-34 network from [6] with a few layers removed and the number of filters per layer reduced by half [5].

FaceNet

FaceNet model is described in [9]. The architecture implemented in [10] is an Inception-ResNet with an embedding of dimension 512. It reaches 99.65% on LFW benchmark.

In contrast to traditional network approaches, which train a classifier and then use an intermediate bottleneck layer as face representation, FaceNet directly optimize the embedding itself using a triplet based loss function.

Figure 2.2 from [9] shows how triplet loss works. Each triplet consists of two matching faces and a non-matching face and the loss aims to separate the positive pair from the negative one.

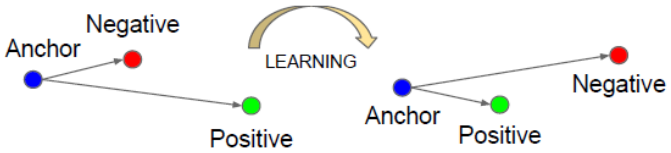


Figure 2.2: Example of a triplet loss

The correct choice of the triplets to use is relevant for achieving good performance. Ideally, hardest positive and hardest negative samples should be selected. In practice, positives and semi-hard negatives are used.

Chapter 3

Implementation

3.1 Introduction

The main purpose of this work consists in the creation of a system, named *Smart Mirror*, which, once received two images in input (a document and a selfie of the user), is able to establish whether the user is the person the document belongs to.

3.2 System architecture overview

Software architecture is represented in figure 3.1.

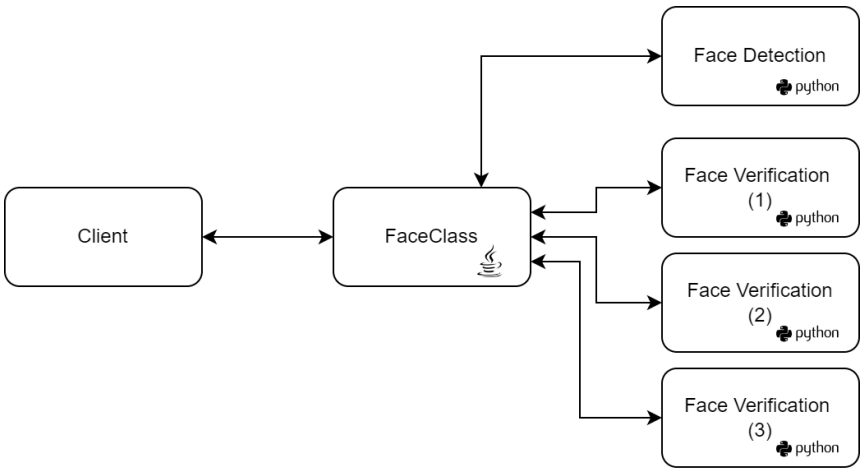


Figure 3.1: Smart Mirror architecture

Two main components can be distinguished: the main application and the microservices.

Main application

This application is the entry point of the system and the only component the client should interact with. It exposes two services, one for face detection and the other for face verification.

However, it does not compute any operation on the images: it simply receives requests from the client and forwards them to the appropriate microservice(s). Then, it elaborates their responses and returns a final result to the client.

Since Java is the main programming language used by the company, this software is written in Spring Boot to ensure easy maintainability and interoperability with other company applications.

Microservices

These applications are supposed to interact only with the main application.

Each of them exposes only one service and is able to solve a specific task (face detection or verification) by using the appropriate model or library.

Since artificial intelligence libraries and models have higher support for python, these microservices are written by using this language and Flask.

3.3 How it works

3.3.1 Face detection

Face detection consists in finding a face inside an image. Given an image, this implementation returns the location of the most probable detected face (if any).

Confidence threshold

A detected face is returned only if its confidence is greater or equal than a confidence threshold. Reducing the threshold may lead to false positives detection, while increasing it will make harder to detect faces.

In order to determine an appropriate threshold value, the detector was tested with a set of identity documents of different type (identity cards, passports, etc). A proper threshold could be set around 0.9 (see section 4.1.2).

Image rotation

Detectors cannot work with upside-down images. To deal with this issue, it is possible to apply one or more rotations to the picture. Each time the image is rotated, the detector is run. At the end, the face with the highest confidence is returned (if any), together with the angle of the rotation.

Face alignment

Face alignment consists in rotating a detected face by a few degrees such that the eyes are aligned, namely they have the same ordinate. Even if face alignment is not mandatory before feeding images to face verification models, it is recommended.

3.3.2 Face Verification

Face verification consists in comparing two faces and establishing whether they represent the same person or not.

Mechanism

To increase the reliability of the result, more than one microservice is requested to verify the two input images, and finally their responses are gathered and elaborated according to the following voting schemes.

Soft voting

As a first attempt, faces are verified if the average of the confidences returned by the microservices is greater or equal than a given (first) threshold.

Hard voting

In case soft voting fails, a second attempt consists in discarding the lowest confidence (among the three returned by the microservices). Among the remaining confidences, the ones with a value greater or equal than a second threshold[1] are considered. If more than half of microservices satisfy this condition, faces are verified otherwise not.

3.3.3 Models

According to the company, the use of more neural networks to perform the same task may guarantee the reliability of the result.

As a consequence, face verification, which is the final objective of this system, uses three models. Face detection, instead, which is a preliminary task to face verification, uses only a single model.

For both tasks, the choice of models fell on available and widely used models with good performance: MTCNN [13] from [8] was selected for face detection, while ArcFace [3] and FaceNet [9] from DeepFace framework [10] and Dlib [4] face recognition model were used for face verification.

These last three models already include their own face detector, which has been disabled to avoid detection from being performed multiple times.

3.3.4 Example of use

An example of interaction between the client and the software components is described in figure 3.2.

[1] The new threshold should be higher than the previous one.

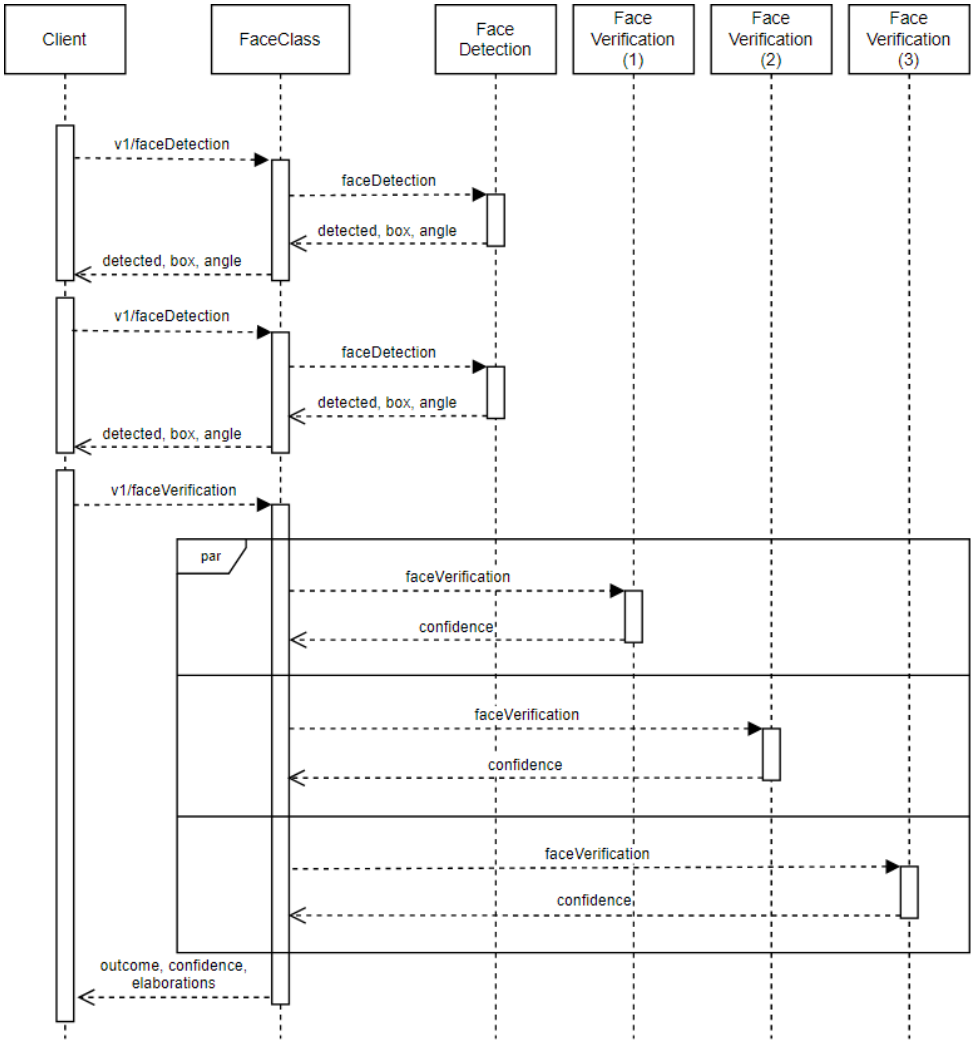


Figure 3.2: Sequence diagram

1. The client sends two (sequential or parallel) requests to the main application, asking for face detection on two pictures (e.g. a document and a selfie);

   (a) the application forwards the requests to the microservice in charge of the face detection;

   (b) the microservice returns, for each request, if a face was `detected` or not, and eventually the bounding `box` and the rotation `angle`;

   (c) the main application forwards the responses to the client;

2. the client should resend a request with a different picture if no face was

detected, otherwise it should:

   (a) crop the pictures (using the received bounding box coordinates);

   (b) eventually rotate/align the pictures (using the provided angle);

3. the client sends a new request to the main application asking for face verification on the cropped (and eventually aligned) pictures;

   (a) the application sends parallel requests to the microservices in charge of face verification;

   (b) each microservice applies an algorithm and returns a `confidence` score;

   (c) the application elaborates the confidence scores, applies a voting scheme and returns the final `outcome`, the `confidence` score and the intermediate `elaborations`.

Chapter 4

Evaluation

4.1 Experimental settings

4.1.1 Datasets

In the experiments, the following two popular dataset are used.

MIDV dataset

This dataset contains photos and videos of documents. There exists more than a single version. The one used in the following is the MIDV-2020 [2], which comprises 10 document types, each present in previously published MIDV-500 and MIDV-2019 datasets. For each of document type, a sample of 100 documents (of different people) is available.

LFW dataset

Labeled Faces in the Wild (LFW) [7] is a dataset of face photographs designed for studying the problem of face recognition. The dataset contains more than 13,000 images of faces collected from the web. Each face has been labeled with the name of the person pictured. 1680 of the people pictured have two or more distinct photos in the dataset.

4.1.2 Face Detection

In order to determine an appropriate threshold value, the detector was tested with a set of identity documents of different type (identity cards, passports, etc.) from MIDV dataset.

Confidence and image resolution

First, images from the MIDV dataset were resized multiple times in order to evaluate the behaviour of the detector with images of lower resolution and, consequently, lower quality.

Starting from images at their original dimensions[1], they were progressively reduced, in such a way the smaller dimension[2] was almost halved, matching the values: 1000, 500, 200, 100. The aspect ratio was maintained in order not to alter the content of the picture.

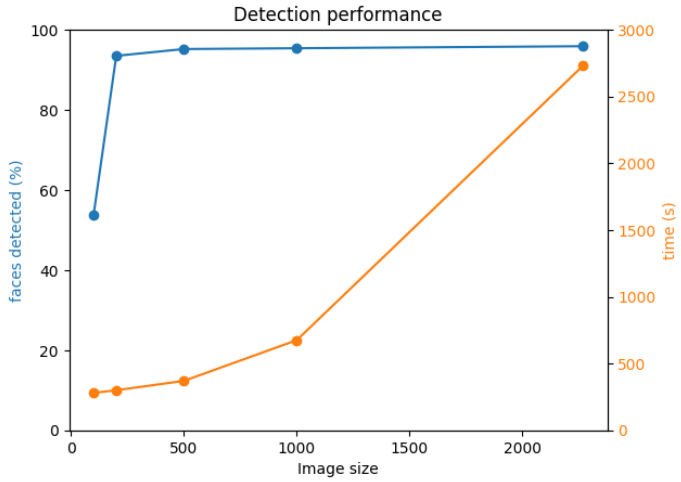


Figure 4.1: Face detection performed at different resolutions

Figure 4.1 shows the percentage of faces detected and the amount of time required at different resolution levels.

As expected, the required time grows as the resolution increases. Also the amount of detected faces varies according to the resolution. However, it

[1] All the images belonging to the MIDV dataset have dimensions 2260 x 4032 pixels
[2] Since all images are in portrait mode, the smaller dimension is always the width.

should be considered that both the amount of detected faces and the confidence at which they are detected is only marginally affected by the change of the resolution: firstly, the percentage of faces detected is almost constant (except for the first case, in which the amount of faces collapsed); secondly, the confidence of each face is substantially unchanged.

From the figure above, it could be concluded that the second resolution is the most efficient. However, in the following, the third one is used, because it retrieves almost all the faces detected at the highest resolution at a low additional cost and better preserves the quality of the image.

Confidence threshold

Since the face detector returns, along with the coordinates of the found face, a confidence value, it is required to establish a threshold to distinguish which of the input pictures contain a face and which, instead, should be rejected.

Also in this case, experiments were run on the MIDV dataset. To further investigate the possibility of false detections, each document was rotated four times, by 90° at each time.

In addition to the already detected 951 faces, 49 false positives[3] were found. Figure 4.2 shows the confidence of true and false positives.

From the figure it emerges that:

- almost all faces were detected with a truly high confidence (99%);
- false positives were detected with a value lower than 95%[4];
- there were no detections with confidence lower than 70%.

According to the severity of the detection, two possible solutions could be suggested. First, the threshold is set to a value close to 99%, to ensure only true positives are selected. Alternatively, it it possible to adopt a threshold around

[3] Upside down faces were not included.
[4] Except an observation with a value of 96%

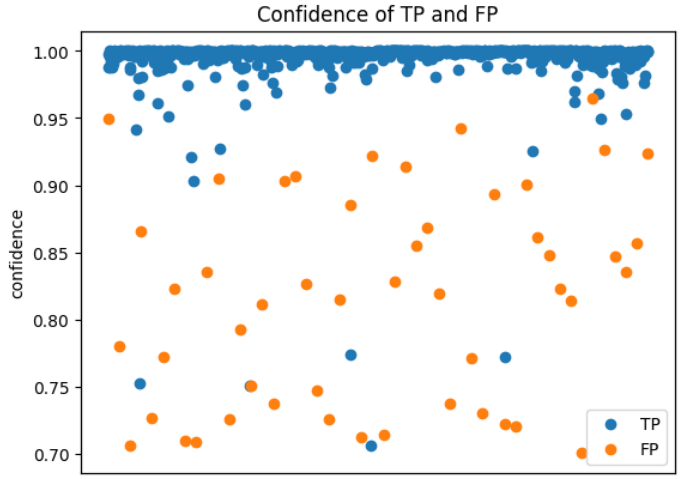


Figure 4.2: Confidence of true and false positive face detections

90-95%, which will include additional true positives at the cost of acquiring also some false positives. An even lower threshold is discouraged.

In this application, face detection is relevant but at the same time is a preliminary step to the final objective, consisting in face verification. Any false positive will surely be rejected in the successive step. For this reason, the threshold was set to 0.9.

4.1.3 Face Verification

The need for a confidence value

A "limitation" of the face verification models that were chosen consists in the fact that they return only a boolean value, indicating the result of the verification. In most of the cases, also a confidence value would be desirable.

In fact, it is particularly relevant to distinguish between truly verified images and those whose result is borderline. In this case, the approval or the rejection of similar images containing the same person may be affected not only by the face of the person itself, but also by other elements such as the quality of the image, and may lead to different results.

True and false matches distributions

A method to establish a confidence value consists in finding the distribution of true and false matches, and then computing to which distribution the new item is more likely to belong to.

To obtain a population, a subset of pictures from Labeled Faces in the Wild (LFW) was used. First, people with at least 10 photos per each were selected, then, 100 identities were random sampled among them. This ensures the possibility of having a total of 4500 true matches. After that, an equal number of false matches was sampled.

After running face detection on this subset, for each face verification model, embedding of the input images were computed.

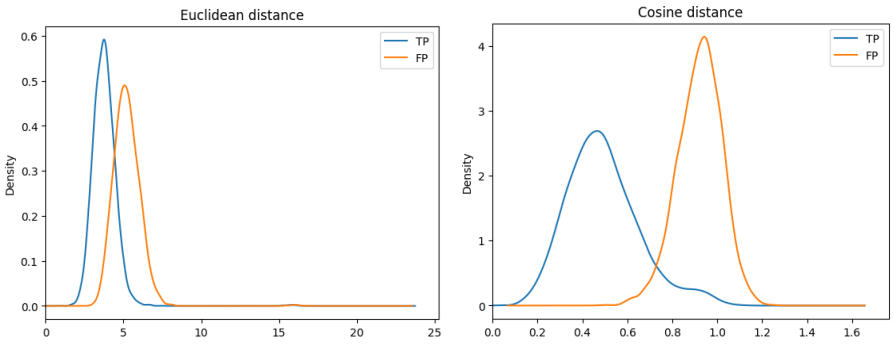
Then, distances between each pair of images in true and false matches were calculated. These distances were used as population to obtain the distribution of true and false matches. Since the aim of face verification consists in the distinction among these pairs, it is desirable to have distribution curves as separated as possible.

Distance metric

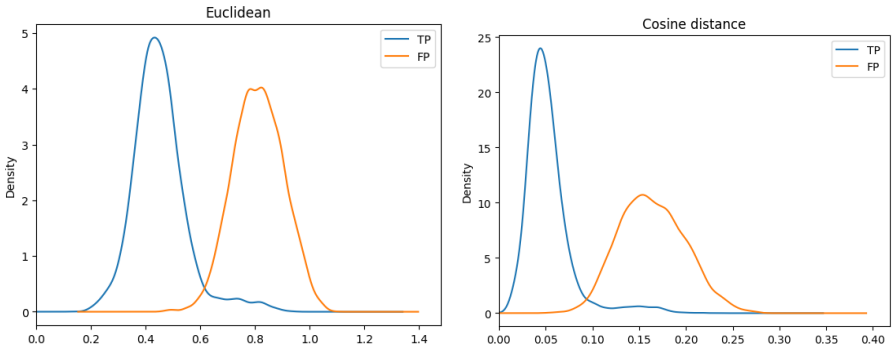
In order to measure the distance between each pair, distance metrics need to be defined. A traditional metric is represented by the euclidean distance, but, in general, cosine similarity is preferred to it.

Here, the appropriate metric for each model was obtained comparing the curves of the distributions, as in figure 4.3.

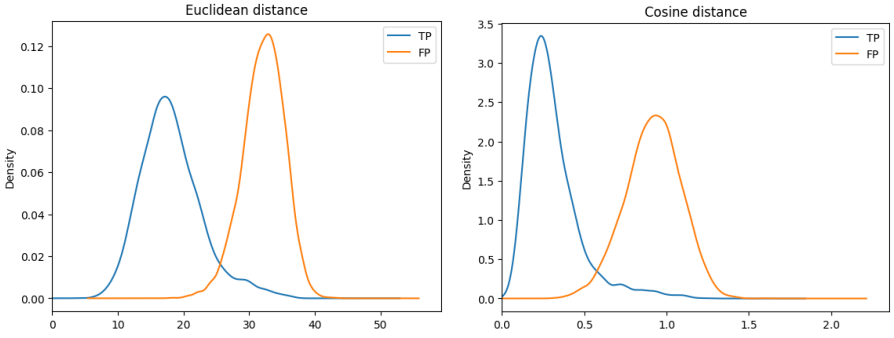
While for ArcFace and Dlib models it seems evident that, respectively, the cosine distance and the euclidean distance guarantee better separability of the curves, for FaceNet model the level of separation is similar. So, for this model, cosine distance was used since generally preferred.



(a) Euclidean and cosine distances for ArcFace model



(b) Euclidean and cosine distances for Dlib model



(c) Euclidean and cosine distances for FaceNet model

Figure 4.3: Distributions of true and false matches with different distance metrics.

Confidence estimation

Given the distribution curves for each model, the confidence was computed as follows:

$$confidence(d) = \frac{y_{true}(d)}{y_{true}(d) + y_{false}(d)}$$

where d is the distance between a pair of images and y_{true}, y_{false} represent the probability of d of belonging, respectively, to the true and false matches distribution.

Confidence threshold

Once the confidence of each model was obtained, their average was computed.

To better contextualize the meaning of the average confidence value, True Positive Rate (TPR) and False Positive Rate (FPR) for different values of average confidence are computed, as can be seen in figure 4.4.

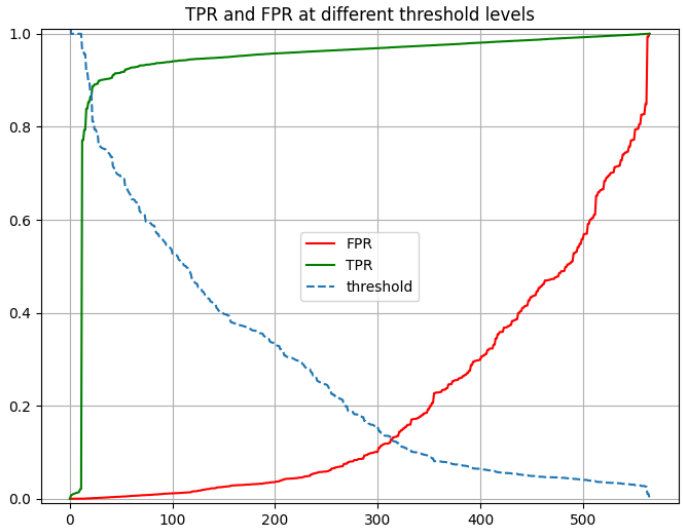


Figure 4.4: TPR and FPR at different confidence threshold levels.

For instance, with an average confidence of 0.7, the TPR is 0.915 and the FPR is 0.004, while for a confidence of 0.5, TPR is 0.953 and FPR is 0.013.

High thresholds ensure that only true positives are detected, but about 10% of them may be missed. Lowering the threshold will include further TP but also false positives. However, even with a threshold of 0.5, the amount of FP is limited.

Again, the choice of the average confidence threshold to adopt depends on the severity of the context where the face verification system is applied.

4.2 Further evaluation

The overall system was evaluated also on a custom dataset, containing images of documents and photos from family, friends and colleagues.

There are 28 identities, with a different number of pictures for each, for a total amount of about 400 pictures. People were asked to pose in several positions and to include also not clear pictures of documents (e.g. warped, at different distances from the camera, in various lightning conditions and from diverse perspectives) with the objective of evaluating the overall system in challenging situations.

4.2.1 Face Detection

With respect to face detection, results were satisfying: almost all faces were detected, with the minimum threshold fixed at 0.9. No false positives were retrieved.

Detected images included even people with sunglasses. Images where detection failed did not include a proper face: people were not showing their face at all (e.g. they were with their back turned), the face was missing because the image represented the back of the document, or the photo was highly blurred.

4.2.2 Face verification

In order to test to face verification, combinations of true and false matches were computed, for a total of about 3000 matches.

On this dataset, the system performed better in the rejection of false matches than in the approval of true ones, as shown in figure 4.5. However, as already stated, the dataset explicitly contains low quality photos that make verification harder.

An alternative that seems to adjust the performance consists in the removal of the minimum confidence (among the three ones considered) when computing the average confidence.

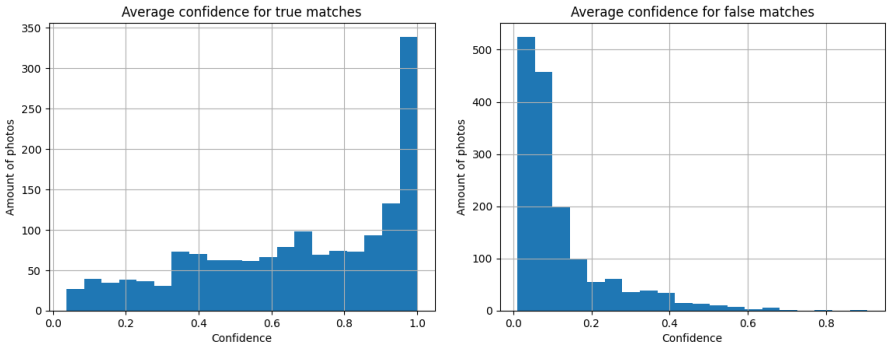


Figure 4.5: Average confidence values for true and false matches

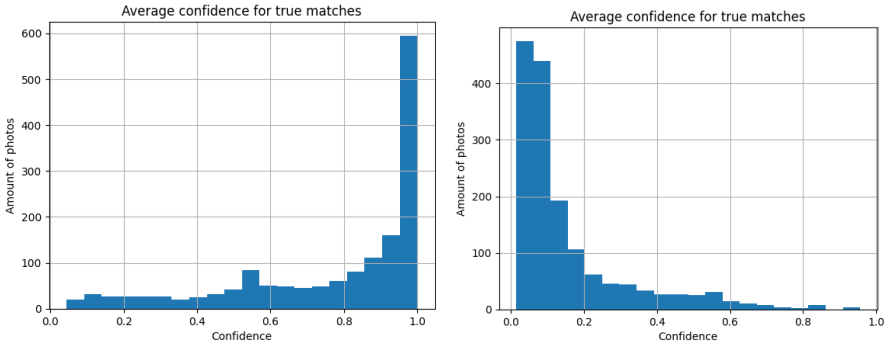


Figure 4.6: Average confidence values for true and false matches, after removal of the lowest confidence

The idea is that the performance of a single model could be low on a specific image, for some reasons. Obviously, the removal of the lowest value will increase the average confidence and this will lead to an increment in the true positives detection. The drawback may consist in the rise of false positives, also.

Experimentally, this idea seems working on the input dataset, as shown in figure 4.6: the amount of true positives is increased, while the one of false positives is almost the same.

Chapter 5

Conclusions

5.1 Final remarks

This work explored the opportunity of an application of artificial intelligence concepts to business related scenarios.

Existing solutions (face detection and verification models) have been inserted in a web application that will be used by a real company.

However, these solutions cannot be simply used “as they are”, but a comprehension of their mechanism is required. In this specific case, the computation and the insertion of confidence thresholds make their results clearer to the final user.

It would be desirable to make further evaluations on more appropriate datasets. However, identity documents data are not free available, at the moment, due to privacy reasons.

5.2 Future work

Here, the provided solution does not take into account the possibility of fake identity documents or user photos.

Establishing the validity of the documents competes to the document recognition application, at which this solution will be integrated. Instead, the insertion of an anti-spoofing filter would be recommended to perform the liveness detection of the user behind the monitor.

Bibliography

[1] Blue Reply: Technology Consulting. URL: https://www.reply.com/blue-reply/it/.

[2] K. Bulatov, E. Emelianova, D. Tropin, N. Skoryukina, Y. Chernyshova, A. Sheshkus, S. Usilin, Z. Ming, J.-C. Burie, M. Luqman, and V. Arlazarov. MIDV-2020: A Comprehensive Benchmark Dataset for Identity Document Analysis. *Computer Optics*, 46(2):252–270, April 2022. DOI: 10.18287/2412-6179-co-1006. URL: https://doi.org/10.18287%5C%2F2412-6179-co-1006.

[3] J. Deng, J. Guo, J. Yang, N. Xue, I. Cotsia, and S. P. Zafeiriou. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*:1–1, 2021. DOI: 10.1109/tpami.2021.3087709. URL: https://doi.org/10.1109%5C%2Ftpami.2021.3087709.

[4] Dlib C++ Library. URL: http://dlib.net/python/index.html.

[5] Dlib models. URL: https://github.com/davisking/dlib-models.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition, 2015. DOI: 10.48550/ARXIV.1512.03385. URL: https://arxiv.org/abs/1512.03385.

[7] 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. Technical report 07-49, University of Massachusetts, Amherst, October 2007.

[8] MTCNN-OpenCV. URL: `https://github.com/linxiaohui/mtcnn-opencv`.

[9] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015. DOI: `10.1109/cvpr.2015.7298682`. URL: `https://doi.org/10.1109%5C%2Fcvpr.2015.7298682`.

[10] S. I. Serengil and A. Ozpinar. LightFace: A Hybrid Deep Face Recognition Framework. In *2020 Innovations in Intelligent Systems and Applications Conference (ASYU)*, pages 23–27. IEEE, 2020. DOI: `10.1109/ASYU50717.2020.9259802`. URL: `https://doi.org/10.1109/ASYU50717.2020.9259802`.

[11] P. Viola and M. J. Jones. Robust Real-Time Face Detection. *International journal of computer vision*, 57(2):137–154, 2004.

[12] M. Wang and W. Deng. Deep Face Recognition: A Survey. *Neurocomputing*, 429:215–244, 2021. ISSN: 0925-2312. DOI: `https://doi.org/10.1016/j.neucom.2020.10.081`. URL: `https://www.sciencedirect.com/science/article/pii/S0925231220316945`.

[13] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, October 2016. DOI: `10.1109/lsp.2016.2603342`. URL: `https://doi.org/10.1109%5C%2Flsp.2016.2603342`.

Acknowledgements

I would like to thank my supervisor Dr. Luigi Di Stefano for his assistance and advice with this thesis.

I would also like to express my gratitude to Drs. Fabio Fasano, Gennaro Orizzonte, Filippo Bregoli and Enzo Pio Palmisano and all the Reply staff for allowing me the opportunity to attend the internship in the company, for providing me guidance throughout this project and for welcoming me into the Reply team.

Finally, I could not have undertaken this journey without the support of my family and friends.