UNIVERSIDADE D
COIMBRA

David Alexandre Mendes Carreira

DEEP FACE RECOGNITION FOR ONLINE STUDENT IDENTIFICATION

Thesis submitted to the University of Coimbra in fulfilment of the requirements of the Master's Degree in Engineering Physics under the scientific supervision of PhD David Portugal and MsC José Faria and presented to the Physics Department of the Faculty of Sciences and Technology of the University of Coimbra.

June 2023

List of Figures

List of Tables

Contents

Acknowledgments

Abstract

Resumo

Chapter 1

Introduction

1.1 Context

The outbreak of the COVID-19 pandemic tested the entire world on several levels and changed the concept of what is ”normal” thereafter. The devastating health, economic and social consequences that COVID caused, spanned a need to develop novel solutions, for almost every aspect of our lives, that facilitate the adaptation to the new world we're living in.

Educational systems were no exception. In the midst of the pandemic, governments around the world forced institutions to shut down and stop the customary in-person regimen of teaching. By April 2020, most universities transitioned to an adapted remote learning [137] that lacked proper support due to the unanticipated nature of the events, leading to new challenges, in particular, the legitimacy of moments of evaluation performed remotely. To counter this problem, different approaches can be taken, namely, changing the method of evaluation, suppressing it altogether [8] or, when possible, implement a continuous monitoring solution such as TrustID [35]. However, there are still unresolved issues that must be addressed in order to implement an end-to-end solution capable of assuring the success of such systems.

One core aspect of them is the face verification stage, where the visual data obtained from the monitoring system directly influences the rate of success of said stage. Other challenge is, due to the purpose of the application and expected devices to be used, what is obtained can be classified as from an unconstrained nature. Therefore, even though the capture of image is consensual, there is no way of controlling the conditions of capturing the visual data and consequent results. This can be attributed to the fact that it is anticipated that the system will be executed in a laptop or a smartphone, thus the capture device might not be ideal. The more probable input method will be a webcam or the smartphone's front facing camera, so a high variation in pose, resolution, illumination, etc. is not unforeseeable.

Another detail that must be regarded, is the processing power available to execute the system[1], since solutions that benefit from higher accuracy comes at the cost of increased computational overhead, which can make real-time continuous monitoring unfeasible.

In conclusion, the method of choice must take the aforesaid into consideration and be a trade-off between accuracy and computational strain, while also being invariant, to a certain degree, to the posed challenges of capturing the required data.

1.2 Objectives

Building upon the earlier context, the main objective of this dissertation is to evaluate face recognition methods and compare them to the TrustID project's student monitoring solution. The aim is to identify the model that offers the most favorable balance of accuracy and computational efficiency. To achieve this, the following specific objectives have been established:

[1] According to the February 2023 Steam hardware survey, roughly 5% of its users do not have a dedicated GPU.

- Conduct a comprehensive review of state-of-the-art face recognition methods and datasets.
- Implement the essential stages of a face recognition pipeline.
- Employ transfer learning techniques using relevant datasets.
- Analyze and assess the performance of the implemented methods using selected benchmarks.

1.3 Dissertation structure

This dissertation will be divided into different chapters that partitions themselves into sections and subsections. Chapter one relates to the introduction of the dissertation, it will present the context and motivation behind the problem and structure of the document. The document continues to the second chapter, it starts with an overview of the History of AI, carries out a survey about the topic's State-Of-The-Art, presented and summarized through the step-by-step analysis of the pipeline of a Face Recognition system, and ends with a comparison table of the discussed methods. In chapter number three, the implemented methods and experiments are described. The forth chapter will present and discuss the results. Finally, chapter five, will draw conclusions of the work achieved in the past several months and prospects for the future.

Chapter 2

State of The Art

2.1 History of AI

The following sections present a broad overview of the history of Artificial Intelligence (AI) by presenting important articles in order for the reader to be able to have a notion of the progress that has been made over the past decades, the hardships encountered and how important AI is in our lives.

Philosophy

On October 1950, in his article *Computing Machinery and Intelligence*, Alan Turing questioned: "Can machines think?" [122]. At the time, the question was too meaningless to answer since not only the theory but also the technology available weren't developed enough. Nonetheless, Turing still predicted that in the future there would be computers that could, effectively, display human-like intelligence and discernment under the conditions proposed on the aforementioned article.

Relevant events to the birth of AI

The breakthroughs of AI are predominant, and its importance in our everyday life is undeniable, but the theory behind it has several early roots. The interest in

the area grew immensely with, for example, all the Turing's theoretical research, the proposal of the first mathematical Artificial Neuron model in 1943 by Warren McCulloch and Walter Pitts (based of binary inputs and output) [83] and in 1949 Donald Hebb revolutionized the way the artificial neurons were treated by proposing what is known as the Hebb's rule[1]. Taking into consideration the latter two, but specially Hebb's proposals, Belmont Farley and Westley Clark implemented in 1954 one of the first successful Artificial Neural Networks (ANN), also called Perceptron, composed of two layers of 128 artificial neurons with weighted inputs [36]. Over the span of approximately ten years, multiple researches were performed attempting to computerize the human brain. However, only in 1956, during the *Dartmouth Summer Research Project on Artificial Intelligence* [82], was the term "Artificial Intelligence" firstly proposed by John McCarthy *et al.*, beginning what is now considered to be the birth of AI [149].

The fading of general interest

The succeeding two decades following the Dartmouth conference were filled with important developments, with special emphasis in the works published in 1958 by Frank Rosenblatt (generalized the Farley and Clark training to multi-layer networks rather than only two) [101], the 1959 General Problem Solver implemented by Allen Newel *et al.* (a program intended to work as a universal problem solver that was capable of solving exercises such as the Towers of Hanoi[2]) [89] and the ELIZA a natural language processing tool program developed by Joseph Weizenbaum between 1964 and 1966 [134]. Unfortunately, part of the interest and development around AI met an unforeseen fade after criticisms about the exagger-

[1] "When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency, as one of the cells firing B, is increased." [46], meaning that when two neurons fire together their relation is strengthened.

[2] *The Towers of Hanoi* is a game with 3 stacks of increasingly smaller disks. The goal is to stack them one at a time, so that they are arranged in a decreasing radius manner.

ated public funding [42] and the Marvin Minksy and Seymour Papert 1969 book *The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain* [86] that reported on the problems of the Perceptron network. The overall sentiments regarding this topic of research was of doubt and fear of no progress, mainly due to the spending and two issues raised by Minsky and Papert: the ANN couldn't solve linear inseparable problems[3] and there were limitations due to a lack of sufficient computing power to handle the processing of multi-layer large networks.

A better approach

Minksy and Papert raised important questions, but it shouldn't have discouraged other researchers from further trying, since they failed to acknowledge alternative approaches that had already solved those exact problems. As previously stated, the model proposed by McCulloch and Pitts, later improved by the Farley-Clark implementation and, finally, Rosenblatt, couldn't handle linearly inseparable classes. A possible solution for cases like this started being studied in the 1960s [58, 102] and, although it didn't produce relevant results, in 1965 Alexey Ivakhnenko and Valentin Lapa [56] were, indeed, successful in implementing what is nowadays considered to be the first deep learning network of its kind [107]. In 1971 Ivakhnenko also published an article describing a deep learning network with 8 layers that was already able to create hierarchical internal representations [57].

The years progressed, in 1979 Kunihiko Fukushima introduced the first Convolutional Neural Network (CNN) in a structural sense, due to its similarity to the architecture of modern ones of this category. Ten years later, Yann LeCun *et al.* applied to a CNN, for the first time, a revolutionizing algorithm called Backpropagation [69], creating what is now a pillar for most of the modern competition winning networks in computer vision [107] and employing the term "convolution"

[3] That is, if two sets X and Y in $\mathbb{R}^d$ can't be divided by a hyperplane such that the elements of X and Y stay on opposing sides, then we're dealing with linear inseparable classes [34]

for the first known time [74]. He also introduced the MNIST (**M**odified **N**ational **I**nstitute of **S**tandards and **T**echnology) dataset, a collection of handwritten digits [71], that to this day is still one of the most famous benchmarks in Machine Learning. Backpropagation can be traced back many decades, but the modern version was first described by Seppo Linnainmaa (1970) [75], implemented for the first time by Stuart Dreyfus (1973) [30] and, finally in 1986, David Rummelhart *et al.* popularized it in the Neural Network's (NN) domain by demonstrating the growing usefulness of internal representations [104].

The importance of Convolutional Neural Networks

The study on Neural Networks continued and there were improvements on all types of architectures [47, 135] with special highlight to pioneering Neural Networks processed by GPUs[4] (standard NN in 2004 by [91] and CNN in 2006 by [16]). But there's a well deserved particular attention related to the developments of CNNs due to their great performance in image related tasks when compared to others networks, as proven by LeCun in his 1998 paper [71]. Some relevant examples: in 2003 the MNIST record was broken by Patrice Simard *et al.* [110], achieving an error rate of 0.4% (whereas a non-convolutional neural network by the same authors took the second place with 0.7%); three years later, the same benchmark had a new set low of 0.39% by Marc'Aurelio Ranzato *et al.* [99]; in 2009 a CNN by Yang *et al.* was the first network of this type to win an official international competition (TRECVid) [144]; a GPU implementation of a CNN [23] achieved superhuman vision performance in a competition (IJCNN 2011) in a *German Traffic Sign Recognition Benchmark* with a 0.56% error rate (0.78% for the best human performance, 1.69% for the second-best neural network contestant and 3.86% for the best non-neural method [113]). This last example conjoined with non-convolutional methods [96, 25] and the previously cited [16, 91], reinforces how fundamental GPUs were to further develop neural networks. To supplement

[4] Graphics Processing Unit

even more the importance of CNNs and GPUs, only a year later, Alex Krizhevsky *et al.* proposed a Deep CNN trained by GPUs that was the first one to win the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), achieving an error rate of 15.3% while the second place obtained 26.2% [66].

The year of 2012 was very important for Deep Learning, CNNs and Computer Vision, due to all the attention brought to many researches on this topic after several systems of this kind won image analysis competitions ([22, 24] and the very important previously mentioned [66]), beginning what's considered to be the start of the new wave, we're currently in, of interest in Artificial Intelligence, specially in the aforesaid topics [74].

2.2 Face Recognition - Theory Background

Face Recognition (FR) is a thoroughly debated and extensively researched task in the Computer Vision community for more than two decades [98], popularized in the early 1990s with the introduction of the Eigenfaces [123] or Fisherfaces [92] approaches. These methods projected faces in a low-dimensional subspace assuming certain distributions, but lacked the ability to handle uncontrolled facial changes that broke said assumptions, henceforth, bringing about face recognition approaches through local-features [21, 1] that, even though, presented considerable results, weren't distinctive or compact. Beginning in 2010, methods based on learnable filters arose [146, 72], but unfortunately revealed limitations when nonlinear variations were at stake.

Earlier methods for FR worked appropriately when the data was handpicked or generated on a constrained environment, however, they didn't scale adequately in the real world were there are large fluctuations in, particularly, pose, age, illumination, background scenario, the presence of facial occlusion [98] and many unimaginable more [59]. These shortcomings can be dealt with by using Deep Learning, a framework of techniques that solves the nonlinear inseparable classes problem ref., more specifically a structure called Convolutional Neural Network (CNN) [130].

CNNs are an Artificial Neural Network (ANN) that exhibit a better performance on image or video-based tasks compared to other methods [71]. They were greatly hailed in 2012, after the AlexNet [66] victory, by a great margin, in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC). Just two years later, DeepFace [116] revolutionized the benchmarks scores by achieving state-of-the-art results that approached human performance, reinforcing even further the importance of Deep Learning and shifting the research path to be taken [130].

Given what has been stated so far and the proven robustness, performance, and overall results in computer vision ref. won competitions, the methods discussed

in this dissertation will therefore deal exclusively with Deep Learning approaches. For more information on other methods, please refer to [68].

2.2.1 Convolutional Neural Networks

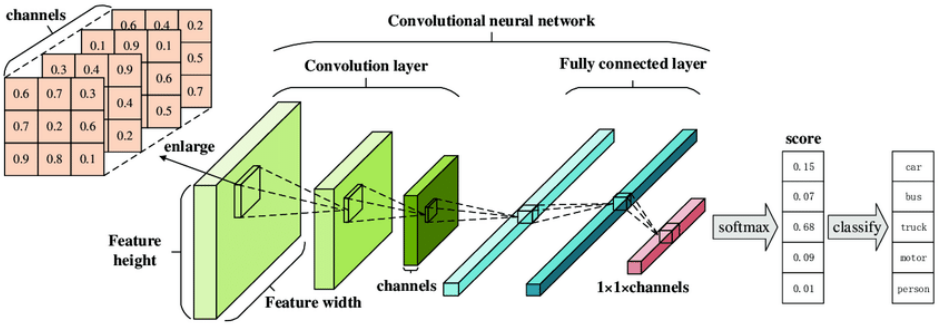


Figure 1: Architecture of a Convolutional Neural Network [60].

There are several types of Neural Networks architectures, but Convolutional Neural Networks (CNNs or ConvNets) are probably the most widely implemented model overall [142, 74]. Using CNNs for Computer Vision tasks [66, 116, 121, 151], in this specific case, Face Recognition, is not an arbitrary choice, but due to the fact that the network design benefits from the intrinsic characteristics of the input data: images have an array-like structure [142], and local groups of values are correlated (motifs or patterns) and invariant to spatial location [70, 14]. Furthermore, when compared to fully connected networks, CNNs are superior due to 4 key features: 1) Shared weights between the same features in different locations reduce the number of parameters [74], 2) sparse connections [3], 3) pooling layers and 4) automatically identifies the relevant features without any human supervision [3, 74].

In the CNN category itself there are different variants, but they all abide the fundamental structure of a feedforward hierarchical multi-layer network (Figure 1). Feedforward because the information only flows in a singular direction without cycling [148], hierarchical because the higher complexity internal representations are learned from lower ones [70, 161] and multi-layer because it is composed of a

series of stages, blocks or layers. The raw data is fed to an input layer, forwarded to a sequence of intercalating convolutional and pooling layers, transmitted to a stage of one or more fully-connected layers [70, 40, 3].

Convolutional Layer

The convolutional layer aims at extracting feature representations from the inputs, and is formed by a set of learnable filters called **kernels** and an **activation function** [40, 142].

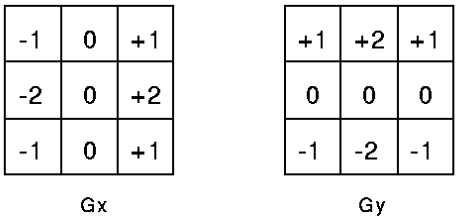


Figure 2: 3x3 Kernels of the Sobel-Feldman operator used for edge detection [112].

A **kernel**[5] (Figure 2), is a grid-like structure of fixed dimensions [WxHxD], where W is the width and H is the height. The dimension D relates to the depth of the kernel and is used to multichannel images, such as RGB ones (3 channels). Each of its elements is a learnable weight that is adjusted during training to extract significant features [3].

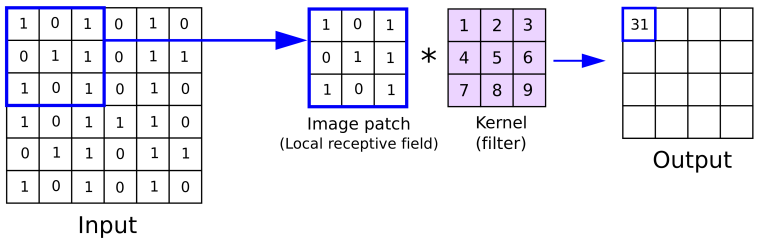


Figure 3: Convolution

[5] Or feature detector [2].

With a predetermined **stride**, the kernel scans its receptive field [61], horizontally and vertically, through the input data, and produces the feature map[6] [70, 3] by performing an element-wise product, called **convolution** (Figure 3), that can be described as follows [61]:

$$f_l^k(p,q) = \sum_c \sum_{x,y} i_c(x,y) \cdot w_l^k(u,v)$$

where $f_l^k(p,q)$ is an element at line p and column q in the feature map from the k-th kernel in the l-th layer, $i_c(x,y)$ is the element at line x and row y in the input data, and $w_l^k(u,v)$ is the weight at line u and column v from the k-th kernel of the l-th layer.

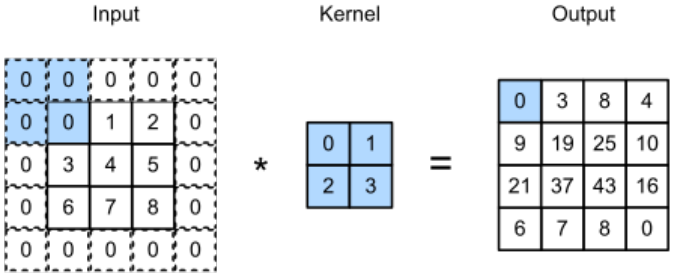


Figure 4: Convolution

At the edge of the input, it's not possible to overlap the whole kernel over all the elements, reaching a point where it can't shift anymore, so the dimensions of the feature map would be unintentionally reduced and information would be lost. Therefore, as seen in (Figure 4), usually **zero padding** is employed to solve the aforesaid problems [142].

The overall architecture of CNNs are inspired by the visual perception [55], so a direct parallelism can be made to better define the **activation function**. The kernels can be seen as receptors, or artificial neurons, that respond to different features, whereas the activation function is a simulation of the threshold function that dictates if the next neuron is activated or not. Additionally, the convolution

[6] Also referred to as feature representation [74] or activation map [2].

operation is linear, consequently, if no nonlinear activation function was used, the input of the next layer would be a linear output of the previous layer. The introduction of nonlinearity through activation functions, such as ReLU (Rectified Linear Unit) and its variations (Leaky, Parametric, Randomized, Concatenated, Bounded, etc) or others like Sigmoid or Tanh [32], allows deep neural networks to approximate any function, enhancing the ability to fit to any data [74].

Pooling Layer

After the features are extracted, the spatial location of them become less relevant for the following layers. Introducing a pooling layer with the purpose of reducing the spatial size of the feature maps by joining identical features [70, 40], then only the dominant response will prevail. This downsampling operation has two important advantages that help reduce the overfitting problem [2, 74]. First, it reduces the number of learnable features, which requires less memory to train the network. Second, it enhances feature extraction invariance to shifts and rotations by emphasizing only the relevant features.

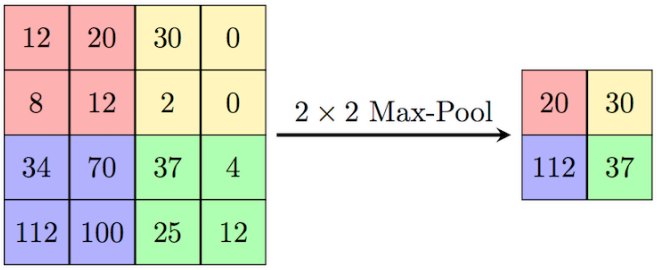


Figure 5: Max pool

There are many ways of downsampling the feature map through pooling such as min pooling, average pooling or stochastic pooling, however max pooling is by far the most popular one. As pictured in (Figure 5), this operation divides the feature map in sections and computes the maximum value in each while discarding the other ones.

Fully Connected Layer

The fully connected (FC) layer is located at the end of the network. It is a dense, feedforward neural network in which every neuron is connected to all other neurons [142, 3]. The final feature map is flattened and transformed in a one-dimensional feature vector and the purpose of the FC layer is using this vector as an input, and act as the CNN classifier by performing high logic reasoning [40].

2.2.2 Training

Training a network in the context of CNNs is the process of finding the optimal kernel weights' values. The training data is passed through the model, the predictions asserted and the distance between them and the expected ones is measured by a loss function. LeCun *et al.* [71] presents the problem as follows:

$$E^p = \mathcal{D}(D^p, Y^p)$$

The loss function, E^p, is a measure of error. It computes the difference between the expected result, D^p, and the predicted result Y^p, where $Y^p = F(Z^p, W)$ is a function of the p-th training example, Z^p, for a determined weight W. Additionally, $E_{train}(W)$, is the average of all the errors calculated in a training set $\{(Z^x, D^x), \ldots (Z^p, D^p)\}$. Training the network is nothing more than a function minimization problem, and for that there are several techniques, the **optimizers**.

Optimizers

By computing the gradient of the loss function in respect to the learnable parameters it allows studying how said parameters influence the loss function and how it can be more efficiently minimized. This algorithm is called **gradient descent** [103], and is the simplest minimization technique. By definition, the gradient of a function is the direction of the steepest ascent, henceforth, the learnable parameters are updated once every epoch in the opposite direction to reduce the

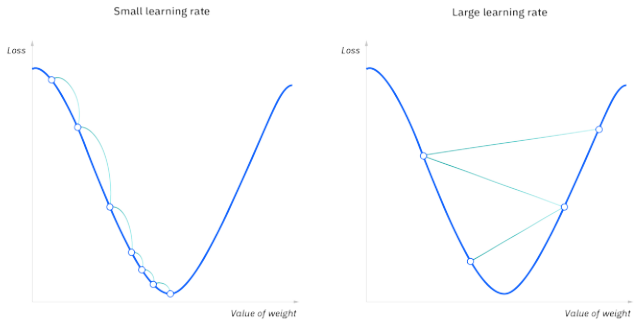


Figure 6: Gradient descent

error:

$$W_k = W_{k-1} - \epsilon \frac{\delta E(W)}{\delta W}$$

where ϵ is a constant called **learning rate** that controls how much the parameters are updated and has an immense impact in the performance of a neural network (Figure 6). According to Goodfellow *et al.* [39], if the learning rate is set too low, training will be much slower and can become permanently stuck with a high training error. On the other hand, if the learning rate is too high, the gradient descent can increase the training error instead of reducing it.

Stochastic Gradient Descent (SGD) [3] is the algorithm of choice to optimize the loss function and its main difference is that the parameters updating routine. First, training samples are randomly sampled in every epoch, then the gradient is calculated for each at a time, and, finally, the parameters are updated accordingly. This is an efficient algorithm in terms of memory usage, but, since the updates occur once at every sample, the convergence can be noisy. Other optimizers worth of mentioning are: **1) Mini-batch Gradient Descent** - divides the training data into non-overlapping batches and the parameters are updated on each one of them; **2) Momentum** [94] - introduces a new hyperparameter to the network called momentum, that accumulates previous gradients, that accelerates the optimizer in the correct direction of minimization and reduces error's oscillation during optimization; **3) Adam (Adaptive moment estimation)** [63] -

uses the first and second moments of the gradients (gradient's mean and variance, respectively) to better optimize the parameters.

Backpropagation

Regardless of the optimizer selected, computing an analytical expression for the gradient is trivial, but evaluating it is a computational heavy task when the aim is to find a minimum with respect to all the parameters in the network. To this extent, the backpropagation (or backprop) algorithm is a solution to do so efficiently [69]. Backprop is usually referred to as a training algorithm, but that is not true. Backpropagation is not a training algorithm, it is simply the method of computing the gradient using a fixed sequence, efficient application of the *Chain Rule of Calculus* $\left(\frac{dz}{dx} = \frac{dz}{dy}\frac{dy}{dx}\right)$, considering that it starts at the final layer, and proceeding backwards [39].

Transfer Learning

Developing a Deep Learning system requires data in large scale [93], specially labeled one. In a general sense, gathering information can be very difficult, but if the domain of study is too specific or not widespread enough, that poses an even bigger challenge. To overcome this solution, based on the psychologist C.H. Judd's theory, a technique called Transfer Learning takes place. It is defined by Zhuang *et al.* [163] as: given a source and target domain, transfer learning is the act of utilizing the knowledge acquired in the source to further improve the performance of the learned decision functions on the target domain. A classic example is the case of learning how to ride a motorcycle. It will be easier for someone who has already learned how to ride a bicycle than it is to someone starting from scratch.

In the context of CNNs, there are 2 ways of approaching the aforesaid problem. By taking a pre-trained CNN and either use it as a fixed feature extractor or fine-tune it. In the first case, an already trained CNN is used, however, the final few layers or the fully connected one is discarded and retrained to a specific task, while

the rest of the network is frozen and used as the feature extractor that will feed the classifier input. The second approach is referred to as fine-tuning a network and, as the name suggests, the source network's parameters are used as a starting point and is entirely retrained using the desired data. It is also common to freeze the first layers since they are responsible to extract the more general, universal features (edges or patterns).

2.3 A Face Recognition System

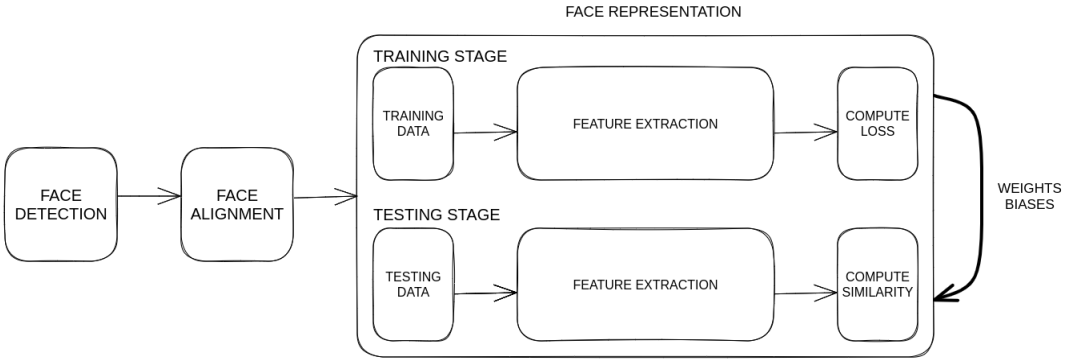


Figure 7: A typical face recognition pipeline, guided by the approach in [130].

According to Ranjan *et al.* [98], the goal of a FR system is to find, process and learn from a face, gathering as much information as possible, and as a result, it is one of the most widely implemented biometric system solutions in light of its versatility when facing real world application [31].

By and large, all end-to-end automatic face recognition systems follow a sequential and modular[7] pipeline (Figure 7) composed of three pillar stages [130]: face detection, face alignment and face representation. First an image or video feed is used as an input then, as the name suggests, the **face detection** module is responsible for finding a face. Next, the **face alignment** phase applies spatial transformations to the data in order to normalize the faces' pictures (or frames, in the case where a video is used) to a standardized view. Finally, the **face representation** stage, makes use of deep learning techniques to learn and further extract discriminative features that will allow the recognition *per se*. A feature is nothing more than a characteristic inherent to the input image, it can be "something" or a collection of, that has been measured or processed, and presented as a result [39].

[7] Sequential because each stage relies on the output from the previous ones, and modular in the sense that each stage employs its own method and it can be modified to better adapt to specific tasks.

All three stages have their individual importance and methods of implementation[8]. **Face detection** is achievable through classical approaches [125, 11] or deep methods, among them is [29] and the widely applied [152]. **Face alignment**, once again, can be accomplished through traditional measures [26, 80] or more modern ones, namely [53] or the aforementioned [152] which concurrently performs detection and alignment. To conclude, the **face representation** module is no exception, and can also be divided in two groups, regarding the methodology used. Some conventional systems were already mentioned, such as [92, 123], and the deep learning ones are the object of discussion of this dissertation and will be reviewed along the following sections, therefore, the focus will be on describing, with particular interest, the face representation stage.

2.3.1 Face Detection

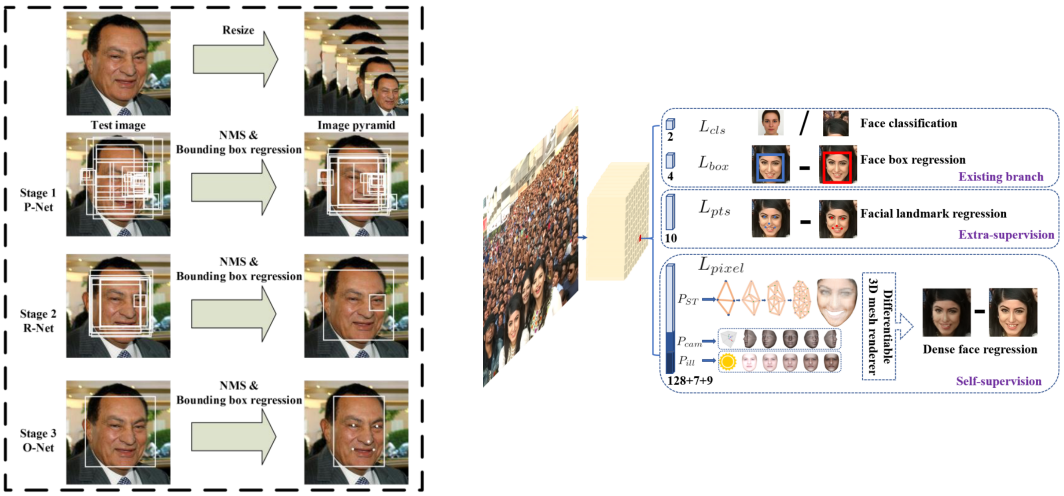
Face detection is the first step in any automatic facial recognition system. Given an input image to a face detector module, it is in charge of detecting every face in the picture and returning bounding-boxes coordinates, for each one, with a certain confidence score [31, 98].

Previously employed traditional face detectors are incapable of detecting facial information when faced with challenges such as variants in image resolution, age, pose, illumination, race, occlusions or accessories (masks, glasses, makeup) [31, 98]. The progress in deep learning and increasing GPU power led DCNNs to become a viable and reliable option that solves said problems in face detection.

These techniques can be included in different categories. A more analytical perspective [31] distributes the methods, depending upon their architecture or purpose of application, over seven categories: multi-stage, single-stage, anchor-based,

[8] For a deeper and extensive study, please refer to: [147] in the case of classic face detection approaches and [85] for deep learning based methods; [132] addresses traditional face alignment methods and is complemented with [31] for more up-to-date techniques; and [68] tackles classic face representation.

anchor-free, multi-task learning, CPU real-time and, finally, problem-oriented. Additionally, being as the face detection problem can be seen as a specific task in a general object detection situation, it is no surprise that several works inherit from them and, therefore, some bases are referenced throughout the next list.



a) b)

Figure 8: Comparison between **a)** MTCNN: multi-stage, CPU real-time and multi-task learning, and **b)** RetinaFace: single-stage, anchor-based, CPU real-time and multi-task learning. MTCNN [152] proposes a series of bounding boxes then, through a series of refinement stages, the best solution and landmarks are found. RetinaFace [29] accomplishes, in a single-stage, face classification and bounding box regression by evaluating anchors, landmark localization and dense 3D projection for facial correspondence.

→ **Multi-stage** methods [29] include all the coarse-to-fine facial detectors that work in similar manner to the following two phases. First, bounding box proposals are generated by sliding a window through the input. Then, over one or several subsequent stages, false positives are rejected and the approved bounding boxes are refined. To complement, one widely applied object detection protocol that inspired face detection methods and perfectly describes the steps mentioned above is Faster R-CNN [100]. However, these methods can be slower and have a more complex way of training [141].

→ **Single-stage** approaches [29] are the ones that perform classification and bounding box regression without the necessity of a proposal stage, producing highly dense face locations and scales. This structure takes inspiration, once again, from general object detectors, for example, the Single Shot MultiBox detector, commonly referred to as SSD [77]. Finally, the methods included in this class are more efficient, but can incur in compromised accuracy, when compared to multi-stage.

→ **Anchor-based** techniques [79, 29, 150] detect faces by predefining anchors with different settings (scales, strides, number, etc.) on the feature maps, then performing classification and bounding box regression on them until an acceptable output is found. As proven by Liu and Tang *et al.* [79], the choice of anchors highly influences the results of prediction. Hence, it is necessary to fine-tune them on a situation-by-situation basis, otherwise, there is a limitation in generalization. Furthermore, higher densities of anchors directly generate an increase in computational overhead.

→ **Anchor-free** procedures, obviously, do not need predefined anchors in order to find faces. Alternatively, these methods address the face detection by using different techniques. For example, DenseBox [52] which attempts to predict faces by processing each pixel as a bounding box, or CenterFace [141] that treats face detection as a key-point estimation problem by predicting the center of the face and bounding boxes. Even so, relating to the accuracy of anchor-free approaches, there's still room for improvement for false positives and stability in the training stage [31].

→ **Multi-task learning** are all the methodologies that conjointly performs other tasks, namely facial landmark[9] localization, during face classification and bounding

[9] A facial landmark is a key-point in a face that contributes with important geometric information, namely the eyes, nose, mouth, etc. [37]

box regression [31]. CenterFace [141] is one example, and so it is the widely implemented MTCNN [152], which correlated bounding boxes and face landmarks. RetinaFace [29] is another state-of-the-art approach, it mutually detects faces, respective landmarks and performs dense 3D face regression.

→ **CPU real-time** methods, as the name suggests, include the detectors that can run on a single CPU core, in real-time, for VGA-resolution input images. A face detector can achieve great results in terms of accuracy, but for real world applications, its use can be too computational heavy, therefore, can't be deployed in real time (specially in devices that do not have a GPU) [31]. MTCNN [152], Faceboxes [154], CenterFace [141] or RetinaFace [29] are examples of this category.

→ **Problem-oriented** is a category that includes the detectors that are projected to resolve a wide range of specific problems, for example, faces that are tiny, partially occluded, blurred or scale-invariant face detection [31]. PyramidBox [119] is an example that solves the partial occluded and blurry faces, and HR [50] tackles the tiny faces challenge.

Although this distribution can create some overlap among the categories, it is superior due to the simplicity of inferring what defines each category and being a more fine-grained way of classifying techniques when compared to others, namely the dual categorical division by [98] that groups the methods in region[10] or sliding-window[11] based.

[10] Region-based approaches creates thousands of generic object-proposals for every image, and subsequently, a DCNN classifies if a face is present in any of them.

[11] Sliding-window approaches centers on using a DCNN to compute a face detection score and bounding box at every location in a feature map.

2.3.2 Face Alignment

Face Alignment, or facial landmark detection [15], is the second stage of the face recognition pipeline, and has the objective of calibrating the detected face to a canonical layout, through landmark-based or landmark-free approaches, in order to leverage the core final stage of face representation [31].

Despite the fact that traditional face alignment methods are very accurate, that only occurs in constrained circumstances. Therefore, once again, to address that issue, deep learning-based methods are the solution to perform an accurate facial landmark localization that realistically scales to real world scenarios [37].

Furthermore, face alignment, can be accomplished through two categories of methods: landmark-based and landmark-free.

→ **Landmark-based alignment** is a category of methods that exploits the facial landmarks with the aim of, through spatial transformations, calibrating the face to an established layout [31]. This can be accomplished through: coordinate regression, heatmap regression or 3D Model Fitting. **Coordinate regression-based** methodologies [37, 76, 152] consider the landmark localization as a numerical objective, i.e. a regression, thus an image is fed to a DCNN and it will output a vector of landmark coordinates. **Heatmap Regression** [28, 139, 17] is different from coordinate regression because, although it is a numerical objective task, the output is not a coordinate vector, but a map of likelihood of landmarks' locations. Finally, **3D Model Fitting** [9, 15, 140] is the category that integrates methods that consider the relation between 2D facial landmarks and the 3D shape of a generic face. The particularity of them is the reconstruction of the 3D face from a 2D face image that is then projected over a plane in order to obtain the landmarks.

→ **Landmark-free alignment**, on the other hand, integrates the approaches that do not rely on landmarks as a reference to align the face, in contrast, these type of methods incorporate the alignment into a DCNN that gives, as a result, an

aligned face [31]. An example of an end-to-end method that does not depend on facial landmarks is RDCFace [157], and it rectifies distortions, applies alignment transformations and executes face representation. Hayat et al. [43] proposes a method that deals with extreme head poses. The process to register faces in an image with high pose variance can be quite challenging and often demands complex pre-processing, namely landmark localization, therefore, to address that, a DCNN is employed that does not rely on landmark localization and concomitantly register and represent faces.

As can be seen from the previous section, this step in the face recognition process can be accomplished, very sporadically, through standalone methods that process the detected face from the previous stage, but generally joint detection and alignment methods (and sometimes even face representation), previously referenced in the multi-task learning definition, are the optimal choice [15].

2.3.3 Face Representation

Finally, Face Representation is the last stage of the Face Recognition process. It is responsible for processing the aligned face from the previous stage and mapping the produced feature representation to a feature space, in which features from the same person are closer together and those that are different stand further apart from each other [31].

According to the literature [31, 73, 98, 108, 130], there's a consensus about how Face Recognition can be performed in two settings of operation: face verification and face identification. This distinction is only made possible due to the approaches available in the Face Representation stage that can leverage one, the other or both.

→ **Face verification**, also referred to as **face authentication**, is a one-to-one match, and it's the action of verifying if the query face matches the identity that's

being claimed. These principles are used in biometric systems such as self-service immigration clearance using E-passport. [73]

→ **Face identification**, also called **face recognition**, is a one-to-many correlation process that compares a query face to a database of faces and associates it to the corresponding match (or matches). A typical use case is to identify someone in a watchlist or surveillance videos. [73]

The overall pipeline comes to a conclusion in this module, however, in reality, it goes further than that. As can be seen in (Figure 7), due to its importance for the face recognition problem, it's highlighted the inherent pipeline of the Face Representation stage, henceforth, it shall be discussed in depth in the next section.

2.4 Face Representation Pipeline

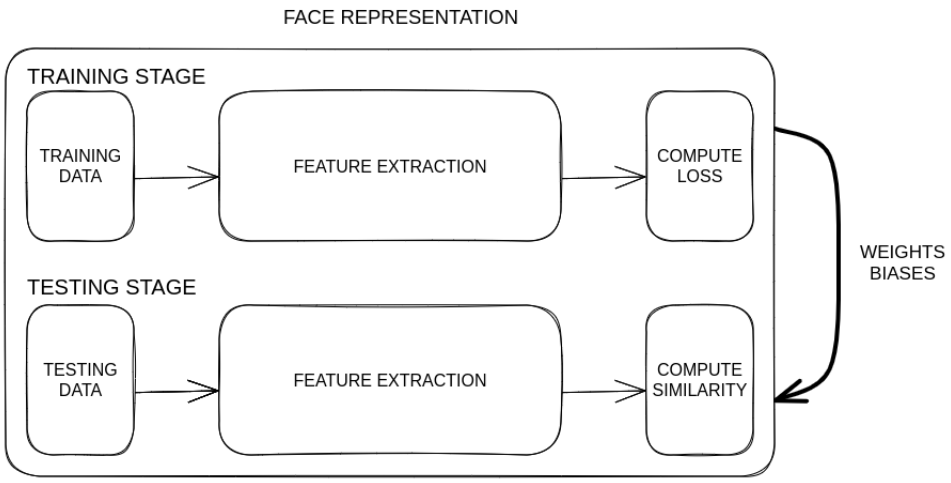


Figure 9: Face Representation pipeline, guided by the approach in [130].

As show in (Figure 9), Face Representation is a two-step module composed of a training and testing stage. So as to be capable of performing face recognition, in either a verification or identification manner, a face representation system needs to learn robust, invariant and discriminative features that can distinguish identities [98].

To meet these requirements, the feature extractor must first be trained properly by taking data from previous stages and outputting a feature representation that's compared to the desired value using a loss function[12] [70, 130]. After that, everything is ready for the testing stage, where the face recognition *per se* occurs by calculating a similarity score for the feature representation produced by the trained feature extractor, and dictating if the identity belongs to the same person (face verification) or if it matches any identity (face identification) [98].

[12] Also referred to as a cost or objective function. [70]

2.4.1 Datasets: Training and Testing Data

As been discussed throughout this dissertation, Deep Learning techniques can solve the problem of handling unconstrained scenarios, where there are variations in pose, illumination, occlusion, and so forth. To support this, in the past few years, datasets have been developed with this in mind so to be able to provide a large and diverse set of both training data, allowing for adequate regularization to unseen circumstances, and testing data that benchmarks the face recognition system in, as similar as possible, unconstrained real world scenarios [31].

In the following pages, a mix of training datasets, of overall relevance and more geared towards the purpose of this dissertation, will be reviewed.

Training Data

When developing a deep face recognition system it's essential to keep in mind its necessity to adapt, and that's where the dataset used for training comes at play. Large training datasets are essential for face recognition [93], but large-scale is not enough. There must be a balance between the depth (number of unique identities) and the breadth/width (number of images per identity) [7, 13], and it will lead to different effects.

On one hand, a training dataset that is deep will help the face recognition system to produce more discriminative feature representations, since it will have a great number of identities to learn from. On the other hand, a wider set will have more images per identity, therefore, variations in pose, expressions, illuminations, occlusions, background clutter, image quality, accessories, and so forth [6] can be introduced and ultimately lead to feature representations more robust to them.

→ **CASIA-WebFace** [145], composed of 494,414 face images and 10,575 identities, was proposed as a novel dataset to overcome the problem of data dependence in face recognition and improve comparability across different methods. By train-

ing on the same dataset, methods can be better evaluated and compared.

→ **VGGFace** [93] was published alongside a homonymous face recognition method and, once again, with the objective of combating the lack of available large scale public datasets. It contains 2,6 million images and 2,622 different identities and a curated version, where incorrect image labels were hand-removed by humans, has 800,000 images for the same amount of identities.

→ **MS-Celeb-1M**'s [41] first intention was to provide a novel benchmark to identify celebrities that solves name ambiguities by linking a face with an entity key in a knowledge base. Second, it aimed at solving the gap in available large-scale datasets by providing a training set with, approximately, 10 million images and 100 thousand identities. Unfortunately, it is a dataset known for the presence of noisy labels.

→ **MegaFace** [88] introduced a benchmark for million-scale face recognition and provided a public large-scale training dataset that integrated 4,753,320 faces over 672,057 identities. The main difference compared to the previously mentioned datasets is that MegaFace does not use celebrities as subjects, in contrast it leverages the photographs released by Flickr under the Creative Commons license.

→ **VGGFace2** [13] is another large-scale dataset, and its main goals are: 1) covering numerous identities, 2) reduce labeling noise through automatic and manual filtering and, finally, 3) represent more realistic unconstrained scenarios due to a novel dataset generation pipeline that gathers images with a broad range of poses, age, illumination and ethnicity. All in all, this resulted in a dataset comprised of 3,31 million faces of 9131 subjects.

→ **UMDFaces-Videos** [7] is a video-based dataset composed of 22,075 videos of 3,107 subjects with 3,735,476 human annotated frames with great variation in image quality, pose, expressions and lightning. It was proposed during a study how

the performance of a face verification models is impacted by the effects of: 1) the type of media used for training (only videos or still images vs a mixture of both), 2) the width and depth of a dataset, 3) the label's noise and 4) the alignment of the faces.

→ **Celeb-500k** [12] is another large-scale proposed with two issues in mind: the disparity in the scale of public datasets when compared with private ones, and determining the impact in performance from intra- and inter-class variations. That being so, Celeb-500k, consisting of 50 million images from 500 thousand persons, and Celeb-500k-2R, a cleaned version of the previous, comprised of 25 million aligned faces of 245 thousand identities, are released.

→ **IMDb-Face** [126] proposes a new dataset with based on a manually cleaned revision of MS-Celeb-1M and MegaFace. The growing demand for large-scale datasets introduced a new variable to take into consideration: the time available to annotate the data. Datasets that are well-annotated and have an enormous amount of data are notably expensive and time-consuming to develop. Therefore, automatic measures to clean the data were used, so it's expected for a certain degree of noise to be introduced in a dataset. After selecting a subset from both the originals datasets, 2 million images were manually cleaned and resulted in 1,7 million images of 59 thousand celebrities.

→ **MS1MV2** [27] is another well know dataset. It was proposed in the ArcFace face recognition method's revision paper and consists of a semi-automatic refinement of the previously mentioned MS-Celeb-1M, resulting in 5,8 million images of 85 thousand identities.

→ **RMFRD** [133] is presented in the context of the need of using a mask, mandated by the COVID-19 pandemic, and that greatly reduces the effectiveness of conventional face recognition methods. Therefore, there was a need to improve

their performance and for that a dataset that provides masked faces is needed. RMFRD pioneered this need by publishing a dataset consisting of 5 thousand masked and 90 thousand unmasked faces from 525 celebrities.

→ **Glint360K** [4] is a training set presented in the Partial FC method paper. It was generated by merging and cleaning the aforementioned Celeb-500K and MS1MV2 datasets, which resulted in 17 million images of 360 thousand individuals.

→ **WebFace260M** [162] takes a giant leap in closing the gap between public available datasets and private ones. Partnered with a time-constrained face recognition protocol, the original paper presented an enormous 260 million faces and 4 million identities noisy dataset, an automatically cleaned, high quality training set with 42 million faces over 2 million identities (WebFace42M), and a smaller scale training dataset derived from the WebFace42M that has 10% of its data (WebFace4M).

→ **DigiFace-1M** [6] is a novel approach that revolutionizes the way of training face recognition models. It is a fully synthetic dataset that proposes mitigating three very relevant problems present in the majority of the conventional datasets: 1) ethical issues, 2) label noise and 3) data bias. The dataset is divided in two parts: part one contains 720 thousand images from 10 thousand identities and part two has 500 thousand images with 100 thousand identities, for a total of 1,22 million images and 110 thousand unique identities.

Dataset	Year	Availability	Images/videos	Depth	Avg. Breadth	Distribution	Description
CASIA-WebFace [145]	2014	Discontinued	494,414/-	10,575	46.7	Public	First public face recognition dataset.
Facebook [117]	2015	-	500M/-	10M	50	Private	*Facebook*'s private dataset used to test different properties in face identification.
Google [108]	2015	-	200M/-	8M	25	Private	Private dataset used to train the FaceNet method.
VGGFace [93]	2015	Discontinued	2,6M/-	2,622	991.6	Public	High width public dataset released alongside VGGFace method.
MS-Celeb-1M [41]	2016	Discontinued	10M/-	100k	100	Public	Large-scale celebrities dataset.
MegaFace [88]	2016	Discontinued	4,753,320/-	672,057	7.1	Public	Non-celebrity dataset.
VGGFace2 [13]	2017	Discontinued	3,31M/-	9131	362.5	Public	High characteristics variation dataset.
UMDFaces-Videos [7]	2017	Discontinued	-/22,075	3,107	7.1	Public	Video-based dataset with great variations.
Celeb-500k [12]	2018	Available	50M/-	500k	100	Public	Noisy celebrities dataset.
Celeb-500k-2R [12]	2018	Available	25M/-	245k	102	Public	Cleaned version.
IMDb-Face [126]	2018	Available	1,7M/-	59k	28.8	Public	Manually cleaned revision of MS-Celeb-1M and MegaFace.
MS1MV2 [27]	2019	Available	5,8M/-	85k	68.2	Public	Semi-automatic cleaned version of MS-Celeb-1M.
RMFRD [133]	2020	Available	95k/-	525	180.9	Public	Dataset of masked and unmasked celebrities.
Glint360k [4]	2021	Available	17M/-	360k	47.2	Public	Cleaned version of the Celeb-500k AND MS1MV2 datasets.
WebFace260M [162]	2021	Available	260M/-	4M	65	Public	Largest publicly available dataset of celebrities faces (noisy).
WebFace42M [162]	2021	Available	42M/-	2M	21	Public	Cleaned and smaller version.
WebFace4M [162]	2021	Available	4,2M/-	200k	21	Public	Smaller version.
DigiFace-1M [6]	2022	Available	1,22M/-	110k	11.1	Public	Large-scale, fully synthetic dataset.

Table 1: Comparison table of the previously described training datasets.

Testing Data

After the training is completed the performance of the system needs to be evaluated on different challenges to properly access if it scales (or generalizes or applies or performs in) to real-world scenarios. A test dataset can be chosen for specific hurdles, for instance, cross-pose, cross-age, racial variations, quality assessment, and so forth [31].

→ **LFW** [51] is the most well-known face verification dataset. It was first released in 2007 as a way of evaluating the performance of face recognition methods, in a verification or pair matching manner, under unconstrained scenarios. LFW divides the dataset in 2 views. View 1 is designed for development, and in the training set contains 1100 pairs of mismatched images and 1100 pairs of matched ones, while the test set has 500 pairs of matched and 500 pairs of unmatched faces. View 2 is intended for performance reporting and splits the data over 10 separate sets, to facilitate 10-fold cross validation, where each one has 300 positive pairs (same identity) and 300 negative pairs (different person), resulting in 6000 pairs. Overall, the dataset has 13,233 face images and 5749 identities (only 1680 persons have two or more images).

→ **YTF** [138] is a video-based benchmark that leverages the greater amount of information provided by a video in comparison to still images. By collecting videos from *Youtube* there isn't an opportunity to control the conditions, hence the footage will support a wider range of characteristic's variation, namely lighting conditions, difficult poses, motion blur, compression artifacts, etc. This resulted in 3425 videos from 1595 identities and a benchmark protocol inspired in the LFW. To evaluate performance, a pair-matching test is designed. From the database, 5000 video pairs are collected, where half are matches and the other half are not, to be divided to allow 10-fold cross validation.

→ **IJB-A** [64] aims at straying further from the saturation in recognition benchmarks by proposing more challenging benchmarks (specifically by including wider geographic distribution and full pose variation) for both verification and identification. It consists of a mix of 5712 images and 2085 videos from 500 individuals, with manual bounding boxes, facial landmarks and, most importantly, labels. IJB-A supports two protocols: search (face identification) and compare (face verification). For both the protocols, the specifications are the same, i.e., ten random training and testing splits are generated using all 500 identities then used to perform sample bootstrapping (instead of cross validation) in order to enhance the number of testing subjects. For each split, 333 subjects are randomly distributed in the training set and the remainder 167 are placed in the testing set.

→ **CFP** [109] studies the effect of extreme pose variations, such as a profile view of a face, in face verification. During collection gender and profession balance, as well as racial diversity, were considered. A number of frontal and profile view images was also set as 10 and 4, respectively. Therefore, after cleaning the initial data, it resulted in 7000 images from 500 subjects. The experimental protocol divided the 500 identities over 10 splits (facilitating 10-fold cross validation) and randomly generated 7 matched pairs and 7 unmatched pairs per identity, resulting in a total of 7000 pairs of faces.

→ **MegaFace2** [88] has already been mentioned in the Training Data section, with exception to detailing the benchmarking protocol. The MegaFace2 is used to train the model that will be tested in 3 datasets: a one million Flickr distraction set, FaceScrub [90] and FG-Net. The first dataset is intended to supply the model with identities that are not to be found (hence being called a distraction set), while the last two contain the known identities to be tested.

→ **CPLFW** [159] is another dataset that tackles the overly optimistic accuracy saturation in classic benchmarks, such as the previously mentioned LFW. To this end, evaluating performance for cross-pose faces of LFW subjects is the matter of study. It contains the same number of 13,233 images of 5749 identities like LFW and the benchmark protocol performance is the LFW *View 2* with some differences: 1) negative pairs are from people of the same race and gender, 2) class imbalance and limited positive pair's diversity is resolved by assuring that each identity has at least 2 images.

→ **CALFW** [160] has the same principles as CPLFW but applied to the age of the subjects (including the negative pairs selection and the class imbalance problem).

→ **AgeDB30** [87], similarly to CALFW, is a dataset that considers the subject's age. It distances itself from other databases by solving the noisy labelling, induced by automatic or semi-automatic methods, by doing so manually. Age-DB has 16,488 images and 568 subjects used in 4 evaluation protocols, similar to LFW's *View 2*, where the main difference between them is the age difference between pairs (5, 10, 20 and 30 years).

→ **IJB-B** [136] builds upon IJB-A and proposes solving flaws that were verified in the previous dataset. First, the improved IJB-B dataset is larger, consisting of 21,798 images and 7,011 videos from 1,845 subjects, with a more uniform racial distribution. Second, the protocols are upgraded due to a greater number of possible

comparisons between images and possible identities.

→ **TinyFace** [19] was presented to fill in the gap of low-resolution face recognition benchmarks with genuine images and not downsampled ones. It is designed for face identification, and is composed of 15,975 labelled images and 153,428 distractors, totalling 169,403 low resolution images, from 5,139 identities. The evaluation protocol is similar to the one used by MegaFace: 1) half of the identities are randomly sampled by the probe set and the other half by the gallery set, and 2) the distractor images are added to the gallery incorporating further complexity to the identification process.

→ **IJB-C** [81] adds 1661 new identities to IJB-B and new end-to-end protocols (to evaluate face detection, identification, verification, clustering) in order to better mimic real-world unconstrained recognition. They have increased diversity, both in geographic location and profession, and occlusion scenarios. IJB-C has 31,334 images and 11,779 videos from 3531 subjects.

→ **IJB-S** [59] is a manually annotated benchmark constructed by collecting images and surveillance videos that presents a challenging face recognition problem. It is a dataset with several challenging variations, namely, full pose, resolution, presence of motion blur and visual artifacts. The aforesaid are tested during 6 different face detection and identification protocols. IJB-S consists of 350 surveillance videos, 202 enrollment videos and 5656 images.

→ **RFW** [131] is a proposed benchmark dataset to evaluate the racial bias of face verification solutions. It is divided in 4 subsets regarding the race of the subjects, where each contains, approximately, 10 thousand images and 3 thousand identities, totaling 40,607 images from 11,429 subjects. The evaluation protocol is the same as the LFW one, but the negative pairs were mined to be difficult and avoid easily saturated performance.

→ **QMUL-SurvFace** [20] is a dataset introduced as a benchmark in the *Surveillance Face recognition Challenge* for face recognition in a surveillance context, and it contains both face verification and identification protocols. By data-mining 17 public person re-identification datasets, it achieves 463,507 facial images of 15,573 identities collected in uncooperative surveillance scenarios. Consequently, it presents a high variance in resolution, motion blur, pose, occlusion, illumination and background clutter.

→ **MDMFR** [124] is brought about in light of the COVID-19 impact, where wearing a mask became mandatory and rendered unusable the traditional face recognition methods. Therefore, in conjunction with DeepMaskNet, MDMFR was released. It is a large-scale benchmark dataset designed to evaluate the performance of both masked face recognition and masked face detection algorithms. The recognition protocol contains 2896 images from 226 identities, intended to benchmark masked face recognition models.

→ **XQLFW** [65] revisits the LFW and modifies it to better evaluate cross-resolution face recognition problems. The evaluation protocol, number of images and identities remains the same (13,233 and 5749, respectively), but the negative pairs are sampled in the same manner as CPLFW [159] and CALFW [160].

→ **CAFR** [158] was introduced in 2022, in the revised paper of the AIM (Age-Invariant Model) as a large-scale benchmark dataset to advance the development of face recognition models invariant to age. It consists of 1,446,500 images from 25,000 subjects and spans a range of ages from 1 to 99 years old. The evaluation protocol divides the data in 10 splits of 2500 pair-wise disjoint subjects, where each one has associated to it 5 matched pairs and 5 unmatched, resulting in a total of 25,000 pairs per split.

→ **FaVCI2D** [95] is face verification benchmark dataset that proposes to address

three relevant flaws : 1) the pairs selected are not challenging enough, 2) the demographics of other datasets are not representative enough of the real world diversity and 3) legal and ethical questions concerning the data used. It is composed of 64,879 images and 52,411 unique identities, where 12,468 are used to create genuine matched identity pairs with balanced gender and geographic distribution.

Dataset	Year	Availability	Images/videos	Depth	Avg. Breadth	Distribution	Description
LFW [51]	2007	Discontinued	13,233/-	5,749	2.3	Public	The most well known face verification public dataset.
YTF [138]	2011	Available	-/3,425	1595	2.1	Public	Face verification video dataset inspired on the LFW
IJB-A [64]	2015	Discontinued	5,712/2,085	500	11.4/4.2	Public	Benchmark that aims at straying from accuracy saturation by proposing a more challenging dataset.
CFP [109]	2016	Discontinued	7,000/-	500	14	Public	Studies the effect of extreme pose variations on face verification.
MegaFace2* [88]	2016	Discontinued	10M/-	100k	100	Public	temp
CPLFW [159]	2017	Available	13,233/-	5,749	2.3	Public	Variation of the LFW for different poses with refined verification pairs.
CALFW [160]	2017	Available	13,233/-	5,749	2.3	Public	Same principles of CPLFW but applied to age related tests.
AgeDB [87]	2017	Available	16,488/-	568	29.0	Public	Similar to CALFW but promotes noise free labelling by doing it manually.
IJB-B [136]	2017	Discontinued	21,798/7,011	1,845	11.8/3.8	Public	Improvement over the IJB-B dataset (more data and more possible pairs).
TinyFace [19]	2018	Available	15,975 (153,428)/-	5,139	3.1	Public	Genuine low resolution face recognition benchmark.
IJB-C [81]	2018	Discontinued	31,334/11,779	3,531	8.9/3.3	Public	IJB-B refinement (more protocols and increased individuals diversity).
IJB-S [59]	2018	Discontinued	5,656/350 (202)	202	28/2.7	Public	Very challenging manually annotated benchmark.
RFW [131]	2018	Available	40,607/-	11,429	3.5	Public	Benchmarks the racial bias of face verification methods.
QMUL-SurvFace [20]	2018	Available	463,507/-	15,573	29.8	Public	Dataset colected in uncooperative surveillance scenarios, presenting high variance of characteristics.
MDMFR [124]	2021	Available	2,896/-	226	12.8	Public	Large scale dataset for masked face recognition.
XQLFW [65]	2021	Available	13,233/-	5,749	2.3	Public	LFW variation to study the effect of resolution on face verification.
CAFR [158]	2022	Available	1,446,500/-	25,000	57.9	Public	Large scale dataset to study the impact of individual's age.
FaVCI2D [95]	2022	Available	64,879/-	52,411	1.2	Public	Face verification dataset that addresses unchallenging pairs, demographic bias and ethical concerns.

Table 2: Comparison table of the previously described test datasets.

Although some of the previously referenced datasets do have a benchmarking component and are described as such, they can also be generally employed to train algorithms. When it is desired to fine-tune[13] a model for a specific challenge that, usually, can be better achieved by employing a "benchmark" dataset that was purposefully developed to evaluate a model's performance on that exact challenge.

A common denominator throughout the machine learning domain, and more specifically deep learning, is a general difficulty in separating concepts with a single line. This is a non-linear science in its nature.

[13] An approach to transfer learning (producing a new model by using a pre-trained as a starting point) [163]. More details on this topic in the section Transfer Learning.

2.4.2 Feature Extractor

A feature extractor is present in both the training and testing stage of the Face Representation process, and comprehends a fundamental part of it, as it is what allows the visual data to be processed and transformed to be evaluated, by transforming the input into low dimensional representations [71]. It is also what distinguishes a Conventional Machine Learning approach from a Deep Learning one.

Being that the following methods are all based in deep learning, therefore, the *modus operandi* abides by the same principles and can be outlined as follows: 1) the feature extractor is a deep neural network, more specifically a Convolutional Neural Network (CNN), that is trained with a loss function, 2) the trained feature extractor contains prior knowledge and is applied on unseen test data and 3) the results are used to compute 1:N similarity (face identification - "who is this person?") or 1:1 similarity (face verification - "are these persons the same?").

Over the years, the architecture of CNNs evolved and became of great importance to all image related tasks. Other than the already mentioned revolutionary AlexNet [66], there are other designs that significantly contributed to breakthroughs and broken benchmark records. There can be general architectures, suchlike VGGNet [111], GoogLeNet [115], ResNet [44], iResNet [33], or specialized architectures. For example, lightweight face recognition implementations such as MobileFaceNet [18], VarGFaceNet [143], MixFaceNet [10] and ConvFaceNeXt [48].

→ **VGGNet** [111] took inspiration from AlexNet and improved its accuracy in image classification by studying the effect of the network's depth. It accomplished that by substituting the 11x11 convolution kernel with stride 4 for a stack of very small 3x3 receptive field with stride 1. The final best performing model was 19 layers deep (16 convolutional and 3 fully-connected) and had 144 million parameters.

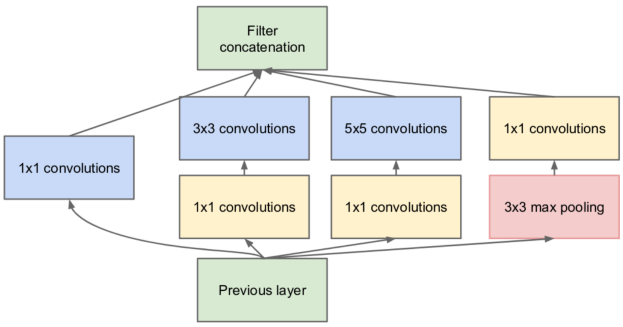


Figure 10: GoogLeNet architecture

→ **GoogLeNet** [115], also known as Inception-V1, aimed at reducing the computational cost associated with training and employing a network, while retaining highly accurate results. That was accomplished by introducing an inception black, consisting of multi-scale convolutional layers with small blocks of different sized kernels (1x1, 3x3 and 5x5). By doing so, it allowed the network to capture information at different scales and increasing both the width and the depth without compromising computational costs. Additionally, sparse connectivity (not all feature maps are shared with the forward layers), replacing the last fully connected layer with a global averaging pooling one and adding a dimension lowering bottleneck layer of 1x1 convolution before employing large kernels, helped to achieve a significantly lower number of 4 million parameters (12x fewer than the revolutionary AlexNet and 36x less than VGGNet). This resulted in its victory on the 2014 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) with a 22 layers deep model.

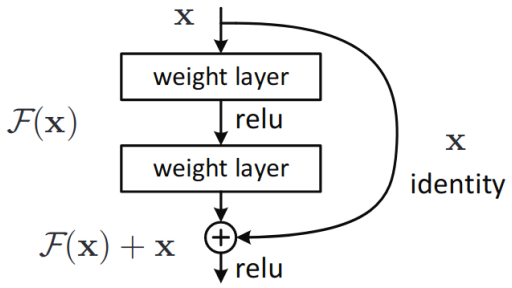


Figure 11: ResNet's residual block

→ **ResNet** [44] is one of the most resourced architecture of feature extractors when applied to face recognition. Its main objective is to efficiently train deeper neural networks that are remarkably more difficult to do so. By reformulating the layers as residual learning functions, it supports deeper architectures while being easier to optimize and, at the same time, producing more accurate results. It also solves the accuracy degradation and the vanishing gradient problems verified when the depth of the network is increased. The main contribution is the introduction of the residual block (Figure 11). It consists of convolutional layers, followed by element-wise addition between the output and the input of the block. This addition performs an identity mapping by creating a direct "shortcut connection" that skips the input directly to the output, allowing the network to learn the difference between the input and the output, i.e., the residual. Enabling the flow of information from earlier layers directly to later layers, facilitates the gradient flow during training and makes it easier for the network to learn. Depending on the depth, there are different variations of the ResNet architecture: ResNet-18 (11.7 million parameters), ResNet-34 (21.8 million parameters), ResNet-50 (25.6 million parameters), ResNet-101(44.6 million parameters), ResNet-152 (60.2 million parameters). The ResNet-152 won the 2015 ImageNet Large Scale Visual Recognition Challenge (ILSVRC), and it was 20 and 8 times deeper than AlexNet and VGGNet, respectively, while being less computational demanding.

→ **iResNet** [33] further improves the ResNet architecture. It proposes: 1) a separation of the network into stages, providing a better path of information flow through the network's layers, 2) an introduction of an enhanced version of the residual learning block with 4 times more spatial channels that focus on the spatial convolution, and 3) an introduction of an improved projection shortcut (used when the dimensions of the previous blocks doesn't match the ones of the next) that includes an additional 3x3 max pooling. A major advantage is that all the improvements above do not increase the original model's parameters and, consequently, overall complexity.

→ **MobileFaceNet** [18] is a set of face verification CNN models designed to perform in real time and with high accuracy on mobile and embedded devices. It is built upon the inverted residual bottlenecks proposed in the general architecture lightweight CNN MobileNetV2 [105] that have the purpose of reducing the number of parameters of the network. The general residual bottleneck block [44] is composed of an input, followed by bottlenecks to reduce dimension and finished with an expansion to restore it, and the shortcut connects the high dimension layers. On the other hand, in the inverted residual architecture, it is considered that the information is stored in the bottleneck layers and the expansion is a mere implementation detail, therefore, a shortcut is placed directly between the bottlenecks. To solve the accuracy problem of face recognition CNNs that have a global average pooling layer, MobileFaceNets replaces it by a global depthwise convolution layer with kernel of size 7x7x1280 followed by a 1x1 convolution as the feature output layer. The primary MobileFaceNet architecture has 0.99 million parameters, but there is also the MobileFaceNet-M (the final 1x1 convolution is removed) and the MobileFaceNet-S (the 1x1 linear convolution before the global depthwise convolution is also removed).

→ **VarGFaceNet** [143] is a lightweight face recognition CNN implmentation based on the VarGNet architecture [153] that introduced a variable group convolution to solve unbalance of computational intensity due to hardware and compilers optimization. VarGFaceNet will use the normal and downsampling blocks from the VarGNet CNN, but will add a Squeeze and Excitation (SE) block [49] and replace the usual ReLU activation function by the PReLU [45] one, since it is better for face recognition tasks. Other than that, the head setting is also changed without losing discriminative ability. The network is started with a 3x3 convolution with stride 1 that preserves the input size, instead of the downsampling 3x3 one with stride 2 from VarGNet. Finally, the embedding setting is also modified by performing variable group convolution and pointwise convolution to shrink the feature map to a 512-dimensional feature vector that's fed to the fully connected layer. However, this shrinking can lead to loss of information, accordingly, to counter that, the number of channels of the feature map is expanded after the last convolution layer, and only then the variable group convolution is performed, decreasing the number of parameters and computation overhead while not degrading the information. In conclusion, the VarGFaceNet network is capable of performing accurate face recognition while maintaining a lower amount of 5 million parameters.

→ **MixFaceNet** [10] is a lightweight implementation of MixNet [118] tailored for face verification. MixNet introduced Mixed Depthwise Convolution Kernels (MixConv) that extends the theory behind depthwise convolution, but employs multiple kernel sizes in a single convolution, promoting the capturing of different patterns from distinct resolutions while reducing the number of parameters needed. The main differences proposed by the MixFaceNet design compared to MixNet resides in the head and embedding settings. For the network head set-up, fast downsampling in the first convolution layer with a 3x3 kernel and stride of 2, and PReLU [45] activation function are used. Regarding the embedding settings,

the global average pooling layer used by the MixNet architecture is replaced by a global depthwise convolution, in the same manner as the previously described MobileFaceNet. Additionally, a shuffle operation is also introduced to the MixConv block. All in all, 3 networks were proposed, MixFaceNet-XS, MixFaceNet-S and MixFaceNet-M, that would be added the prefix "Shuffle" if trained using the aforesaid shuffle operation. Both MixFaceNet-S and ShuffleMixFaceNet-S have 3.07 million parameters.

→ **ConvFaceNeXt** [48] is a family of lightweight face recognition models, based on the ConvNeXt and MobileFaceNets architectures. The original ConvNeXt block is modified to better adapt to face recognition tasks and optimized to reduce the number of parameters. Accordingly, the Enhanced ConvNext (ECN) block is designed by adopting a smaller 3x3 kernel during the depthwise convolution, instead of the original 7x7 one. Adopting the principles studied in the MobileFaceNets implementation, rather than the usual layer of normalization and GeLU in the ConvNeXt model, batch normalization and PReLU [45] are employed. The overall ConvFaceNeXt architecture has 3 main components: the stem partition, the bottleneck partition and the embedding partition. The stem partition is similar to the previously referred "head settings" and is responsible for downsampling the data, and it can be deployed in two settings. The first setting is called patchify head (PH) and utilizes 2x2 convolutions with a stride of two, and the second setting is the typical head (TH) that maintains the stride and changes the kernel size to 3x3. The bottleneck partition is responsible for reducing the number of floating point operations (FLOPs)[14]. Finally, the embedding partition, which is the stage where the face representation (or embedding) is obtained, and instead of average pooling and fully connected layers, once again, global depthwise convolution is engaged. Fine-tuning to the three partitions resulted in 4 different

[14] Not to be confused with FLOPS (Floating point operations per second), a unit of computation speed

networks: ConvFaceNeXt_PS and ConvFaceNeXt_TS, both with 0.96 million parameters, as well as ConvFaceNeXt_PE and ConvFaceNeXt_TE, both with 1.05 million parameters.

2.4.3 Loss

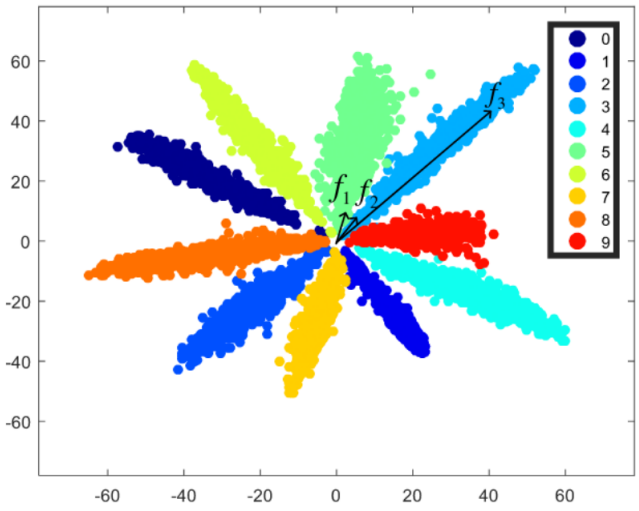


Figure 12: Intra- and Inter-class challenge [128]. Eventhough features f_2 and f_3 belong to the same class, the euclidean distance between f_1 and f_2 is much smaller, proving the ineffectiness of the softmax loss regarding inter-class compactness and inter-class separateness.

The initial face recognition proposals inherited principles from successful object classification implementations, henceforth, the most common loss function utilized was the well-known softmax loss. Unfortunately, it soon proved to be inefficient for face recognition applications since intra-variations (age gap between pictures of the same identity, for example) can be larger than inter-ones (Figure 12). Thus, the investigation interest shifted towards developing loss functions that had a better generalization ability and promoted more separable (to distinguish between an identity) and discriminative (to distinguish between identities) features [130].

Face recognition can be achieved using either metric learning loss functions that learn a feature embedding to compute similarity or softmax-based loss functions

that treat the problem as a classification task. Some works merge the two concepts.

Classification-based loss functions

Classification-based loss functions, derived from the general object classification task, aim at learn an N-way classification of all the classes, where each class relates to an identity composed of several faces [31].

Because the methods are classification-based, the pioneers such as DeepFace or DeepID [114], utilized the most widely implemented loss function for classification, i.e. the softmax loss. It consists of a fully connected layer, the softmax function and cross-entropy loss, and can be formulated as follows:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{W_{y_i}^T x_i + b_{y_i}}}{\sum_{j=1}^{c} e^{W_j^T x_i + b_j}}, \tag{2.1}$$

where N is the number of images, c is the number of identities, y_i is the x_i's ground-truth label, W_{y_i} is the ground-truth weight from x_i in the fully connected layer and b_j is a bias term. The term inside the logarithm represents the probability on the ground-truth class and the training objective is to maximize this probability.

Taking the aforementioned principles and the drawbacks of lackluster generalization, and separable/discriminative abilities, the following loss functions proposed improving the softmax loss to better serve face recognition tasks.

→ **NormFace (2017)** [128] improves the classic softmax loss by studying the effect of L_2 normalization to the features and weights during the training stage. Because introducing this constraint resulted in the network not converging, a scale factor is also adopted that resizes the cosine similarity's scale between features and weights. This normalized softmax loss function can de reformulated as

$$\mathcal{L}_{norm} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\cos(\theta_{y_i})}}{e^{s\cos(\theta_{y_i})} + \sum_{j=1, j\neq y_i}^{c} s\cos(\theta_j)}, \tag{2.2}$$

where s is the scale parameter and $\cos(\theta_j)$ results from the inner product between the L_2 normalized weights W_j and features x_i, i.e., $\cos(\theta_j) = \frac{\langle x_i, W_j \rangle}{\|W_j\|_2 \|i_i\|_2}$

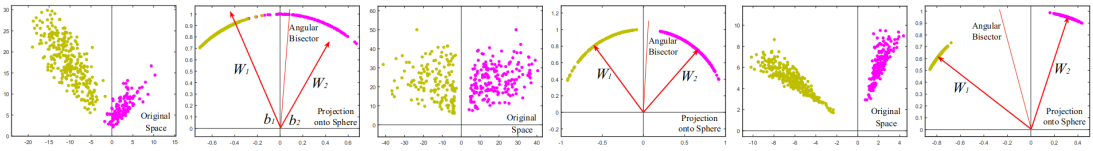


Figure 13: Comparison between the classic softmax loss, modified softmax loss (NormFace) [128] and SphereFace [78].

→ **SphereFace (2018)** [78] improved the intra-class compactness and inter-class distance by introducing a very important concept of angular margin that contrasts with the usual Euclidean margin. As proven by Liu *et al.*, features learned by softmax loss adopt an intrinsic angular distribution, therefore, euclidean margins are not compatible with softmax loss (Figure 14). The decision boundary for a classic softmax loss function is $(W_1 - W_2)x + b_1 - b_2 = 0$, and by normalizing the weights and zeroing the bias, it becomes $\|x\|(\cos(\theta_1) - \cos(\theta_2)) = 0$, where x is a feature vector and the decision will only depend on the angles between class 1 and 2. SphereFace introduces the hyperparameter m ($m \geq 1 \in \mathbb{Z}$) that will, effectively, control the margin size between class 1 and 2 respectively as such $\|x\|(\cos(m\theta_1) - \cos(\theta_2)) = 0$ and $\|x\|(\cos(\theta_1) - \cos(m\theta_2)) = 0$.

→ **AM-Softmax (2018)** and **CosFace (2018)** both improved SphereFace's main problem: the potential unstable training convergence due to the multiplicative angular margin. Thus, an additive cosine margin $\cos\theta_{y_i} + m$ is proposed to facilitate the convergence.

→ **ArcFace (2019)** [27] aims at optimizing the geodesic distance margin since there's a mathematical correspondence between the angle and the arc in the normalized hypersphere. Taking that into consideration, the cosine distance between features obtained by the dot product between the feature produced by the CNN

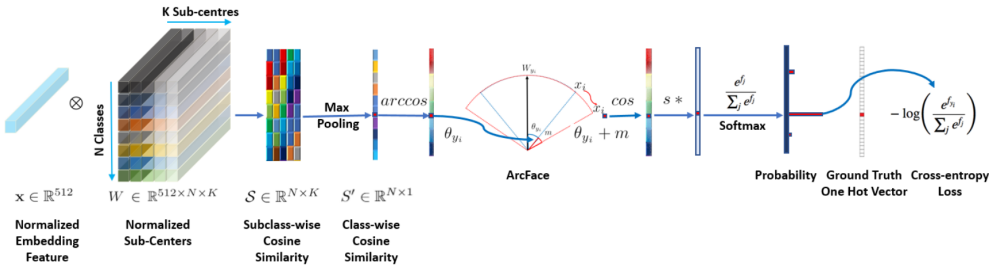


Figure 14: ArcFace [27] training pipeline

and the fully connected layer, then arc-cosine is used to calculate the angle between the feature and the target one. Finally, the authors took the principles from AM-softmax [127] and SphereFace [78] and introduced an additive angular margin directly to the angle $\cos(\theta_{y_i} + m)$, getting the target logit back by the cosine function and further stabilizing the training process and improving the discriminative power of the overall system. The loss function is reformulated by: 1) zeroing the bias and transforming the softmax loss logit $W_j^T x_i = \|W_j\| x_i \| \cos(\theta_j)$, as suggested in [78], where θ_j is the angle between the weight W_J and the feature x_i, 2) following [128, 78, 129] the weights are normalized by L_2, 3) the features x are also normalized in L_2 per [97, 128, 127, 129] suggestion. This normalization insures that there's only an angular dependence between weights and features. By distributing the learned features through a hypersphere with radius s, and combining the margin penalties imposed by SphereFace [78], CosFace [129] and ArcFace, the geodesic distance can be optimized by the following unified framework:

$$\mathcal{L} = -\log\left(\frac{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)} + \sum_{j=1,j\neq y_i}^{N} e^{s\cos(\theta_j)}}\right), \tag{2.3}$$

where m_1, m_2 and m_3 are the hyperparameters relating, respectively, to SphereFace, ArcFace and CosFace.

To this day, ArcFace is still the state-of-the-art regarding face recognition, and one of the most implemented loss functions of this category, consequently, it is usual to see other novel cost functions to try to improve it by using it as a

base foundation to other tasks. The accuracy saturation of more common benchmarks like LFW [51] lead to the appearance of harder ones, namely XQLFW [65] or IJB-S [59], which by itself, ultimately, lead to loss functions aimed at more specific challenges, namely CurricularFace [54], MagFace [84], AdaFace [62] or QMagFace [120].

Metric learning loss functions

Metric learning loss functions include the methods that aim optimizing the distance between feature embeddings. That is, increase the distance between negative embeddings and minimize the distance for those that are positive.

One classic example is the contrastive loss, but this category's attention mainly pends over the triplet loss first implemented by Schroff *et al.* in FaceNet [108].

→ **Contrastive Loss** [31] has the objective of optimizing the distance between identity pairs: positive pairs are encouraged to be closer and negative ones to be further apart. The loss function to be minimized is

$$\mathcal{L}_{contrastive} = \begin{cases} \frac{1}{2}\|f(x_i) - f(x_j)\|_2^2, \text{if} y_i = y_j \\ \frac{1}{2}\max\left(0, m_d - \|f(x_i) - f(x_j)\|_2\right)^2, \text{if} y_i \neq y_j \end{cases} \tag{2.4}$$

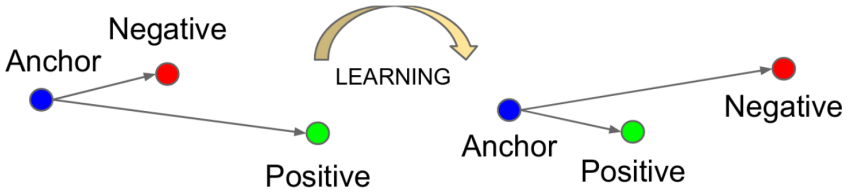


Figure 15: Triplet

→ **Triplet Loss (2014)** [108] embeds an image to a d-dimensional feature vector in an Euclidean space. The motivation is to guarantee that, for every individual i, the squared distance between an image x_i^a (anchor) of an identity and its

corresponding true identities x_i^p (positive) is smaller than the distance between non-identities, x_i^n (negative). This structure is called a triplet (Figure 15). The aforesaid can be described in mathematical terms as:

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2, \tag{2.5}$$

where $f(\dot{})$ are the feature embeddings from all the possible triplets, and α is a margin between positive and negative pairs. Therefore, the loss function to be optimized is:

$$\mathcal{L}_{triplet} = \sum_{i}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\right] \tag{2.6}$$

However, there's a major drawback associated with this methodology. The selection of the triplets are essential for good results and there's a combinatorial explosion regarding the number of possible triplets, specially for large-scale datasets, leading to a slower convergence and computational overhead if all the possible triplets are to be used. The only way to address this problem is through the development of efficient mining strategies that select both hard and representative triplets.

2.5 Related work

The development of solutions for student monitoring has encompassed a plethora of possibilities. Depending on the application, they can be intended for commercial purposes or developed as part of academic research. Commercial solutions such as Kryterion, ProctorExam, ProctorU, Procotorio, ProctorFree, or SMOWL [67] are available; however, with the exception of SMOWL, due to their proprietary nature, their methods of implementation are not disclosed, making it impossible to study them adequately. Therefore, all the methods analyzed, other than SMOWL, will fall under the second category.

→ **Zhang *et al.* [155] (2016)** studied a system based on facial features exclusively. The faces are first detected and extracted using the OpenCV pretrained classifier based on the Viola-Jones/Haar Cascades algorithm. Subsequently, recognition is carried out utilizing the Eigenfaces algorithm, which incorporates Principal Component Analysis (PCA). The main disadvantage of this solution is lacking the flexibility to adverse visual conditions that Deep Learning techniques have.

→ **Atoum *et al.* [5] (2017)** proposed a solution that employs Viola-Jones/Haar Cascades face detector and Minimum Average Correlation Energy (MACE) filter for face verification. Once again, the methodologies utilized are too sensitive to image variations, such as illumination, pose, expressions, etc.

→ **Zhang *et al.* [156] (2018)** latter proposed another solution. In a similar fashion to Zhang *et al.* [155] and Atoum *et al.* [5], the face detection is achieved through the same OpenCV pretrained module. On the other hand, the detected and cropped faces are now processed by a modification of the Stereo Matching algorithm. Because this method extracts information from pairs of stereo images, it is subject to occlusions, ambiguities, textures, etc.

→ **Sawhney *et al.* [106] (2019)** introduced an approach where the face detection is accomplished in two stages: 1) bounding box regression with the Viola-Jones/Haar Cascades algorithm and 2) facial landmark generation with a Local Model-based algorithm. After the faces are extracted, the face recognition stage, in similarity with Zhang *et al.* [155], applies simultaneously Eigenfaces with PCA, therefore, the drawbacks coincide.

→ **Ganidisastra and Bandung [38] (2021)** introduced a Deep Learning-based approach that employs two models for face detection and recognition: YOLO-face for face detection and Facenet for face recognition. Notably, Facenet is continuously trained with data collected at the end of each user session. This training

process causes the system to overfit to each individual identity, leading to reduced adaptability in handling new scenarios due to limited data availability. Furthermore, the training of Facenet utilizes the triplet loss, which can be computationally challenging, especially on less powerful hardware. This is primarily due to the exponential increase in possible combinations when mining triplets during the training process.

→ **SMOWL [67] (2021)** is a multi-modal tool that analyzes voice, face and keystrokes data. For this dissertation, the interest resides only on the facial aspect of it. For face detection, it utilizes a combination of FaceBoxes with MobileNet-SSD for occlusion detection. The face recognition is accomplished with a Facenet model trained with triplet loss on the MS-Celeb-1M dataset. Anew, the triplet loss training can be considered a drawback.

→ **TrustID [35] (2023)** is our solution, designed and developed at ISR - Instituto de Sistemas e Robótica from University of Coimbra. It utilizes as a face detection a linear detector conjointly with a Histogram of Oriented Gradients and pyramidal image search. After the faces are detected and aligned, they're cropped to a Region of Interest (RoI) of 150×150 pixels. Finally, the faces are passed to a face recognition that uses a CNN called ResNet-34, trained with triplet loss on a dataset of, approximately, 3 million faces derived from the Visual Geometry Group Face [93] and FaceScrub [90] datasets.

As stated before, the choice of architecture will be based only on Deep Learning approaches. That is, because the system will be deployed on a difficult scenario where control of, more specifically, illumination, pose and quality will be impossible. Henceforth, a framework insensitive to those variations is the correct choice.

Chapter 3

Methodology

From the previously described systems in Section 2.5, it is possible to infer that a DL approach can be grouped in two implementation choices: the methods of 1) face detection and 2) face representation (that include the feature extracting backbone neural network and the loss function necessary for training[1]). The next sections describe the choices of those and reasons behind them, as well as the implementation and steps necessary for that effect.

3.1 Face detection

Following the proposed pipeline for a Face Recognition system, the initial design choice pertains to the Face Detection module, responsible for detecting, selecting and standardizing the faces. A comprehensive study of solutions was presented in Section 2.3.1, wherein RetinaFace emerged as the best overall solution for a student monitoring context. This approach accepts an image as input and produces multiple outputs, including a bounding box, five key facial landmarks (representing the center of the eyes, nose, and corners of the mouth), and a confidence score that reflects the likelihood of the detection accurately identifying a face. The se-

[1] As this dissertation specifically concentrates on the face recognition domain, the loss function pertains exclusively to the training of that stage.

lection of this option was predominantly influenced by three critical factors: 1) Adaptability to changes in light, pose, facial expressions, etc., facilitated by the DCNN backbone, 2) incorporation of a single-stage approach and leveraging multi-task learning, enabling efficient real-time detection of facial landmarks using just a single CPU core (for VGA resolution), and 3) availability of readily implemented solutions with ample support.

The implementation of choice is a Pytorch implementation[2] of the original code that offers both a MobileNet-0.25 or ResNet-50 as backbones pre-trained on ImageNet. To better suit our application, further development over the original code was needed. RetinaFace is distributed as a general face detection algorithm that's deployed on data with multiple faces to be handled, therefore, there's no built-in methods for face selection or transformations, like alignment and resizing. However, on a student's monitoring scenario, only one face is relevant, and as consequence of that, the face recognition systems are to be trained and validated on single face pictures normalized to a canonical view.

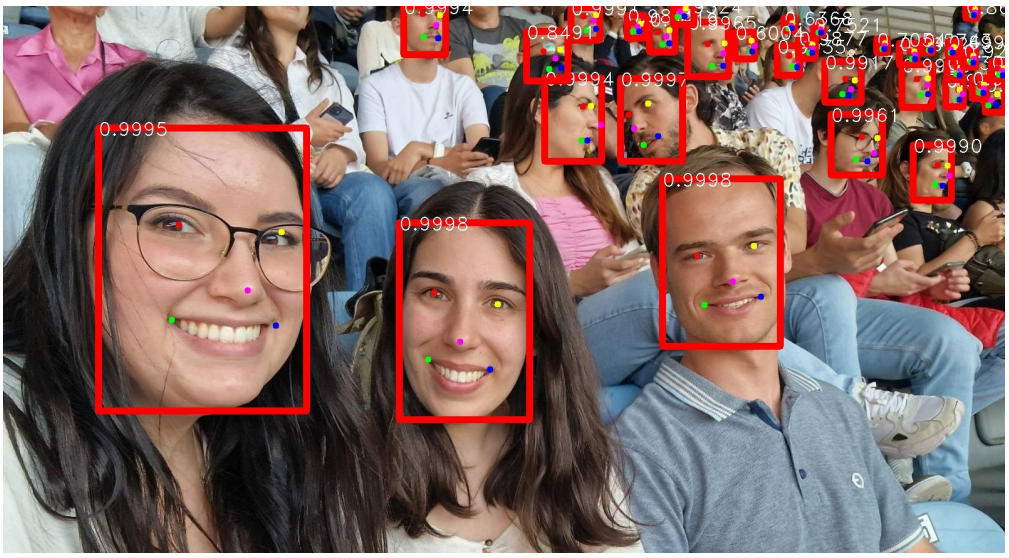


Figure 16: Results produced by the RetinaFace method over a test photo. Represented are the bounding boxes, respective confidence scores and the five facial landmarks.

[2] https://github.com/biubug6/Pytorch_Retinaface

Instantiating the model with default threshold parameters and Resnet-50 as the backbone, it produces the results seen in (Figure 16). From all the bounding boxes available, one must be picked, and one way of doing so is by assuming that the more relevant face is closer to the camera, hence the area of its bounding box will be greater.

Figure 17: Landmark based alignment. The green dot serves as an auxiliary point, resulting from the intersection of a horizontal line originating at the pivot eye and a vertical line projected from the other eye. Subsequently, the rotation angle is determined by computing the arctangent of two distances: the distance between the higher eye and the auxiliary point, and the distance from the auxiliary point to the pivot eye.

The alignment is done with the help of the eyes landmarks as seen in (Figure 17). The eye lower relative to the other is declared as the pivot and starting point of a horizontal line that acts as the reference to calculate the angle of rotation. Following the selection, cropping, and alignment of the face, it is resized to fit the required dimensions for the subsequent phase.

3.2 Face Representation

The Face Representation stage has the task of handling the features of each face and includes a feature extractor and a loss function. Following the review on related works in Section 2.5 Ganidisastra and Bandung [38] and SMOWL [67] both utilize Facenet trained with triplet loss for face recognition. This naturally prompts an investigation into the potential effects on performance when utilizing a more modern network (ResNet variations), as well as a lightweight network with fewer parameters (MobileFaceNet). Furthermore, considering the drawbacks associated with triplet loss training, examining the impact of a well-established loss function, like ArcFace, which encourages intra-class compactness and inter-class distance, would also be noteworthy. Due to time constraints, the methods of choice are all pretrained and subsequently fine-tuned on datasets appropriate to the model's scenario deployment.

3.2.1 FaceNet

For this method, we utilize a highly regarded PyTorch implementation[3] that offers pretrained models in either the CASIA-WebFace or VGGFace2 dataset. The images' faces are cropped and aligned (but not rotated) using MTCNN, and resized to $160 \times 160 \times 3$. However, this implementation has differences compared to the Facenet described in the original paper [108]. Firstly, differing from the original adopted GoogLeNet-style Inception model, the backbone network employed here is the Inception Resnet V1, which integrates residual blocks into the Inception architecture. As highlighted in Section 2.4.2, this modification streamlines the training process by addressing issues such as accuracy degradation and gradient vanishing. Secondly, adhering to the recommendations by Parkhi *et al.* [93], the network was trained as a classifier using softmax loss, departing from the original's triplet loss metric learning approach. Finally, the method's output is a

[3] https://github.com/timesler/facenet-pytorch

512-dimensional embedding, differing from the 128-dimensional output reported in the original paper.

3.2.2 ResNet

The choices for this architecture we're designed to cover a range of neural network depth, complexity and trainable parameters. Henceforth, there are 3 objects of study, ranging from less to deeper ones: iResnet-18 [4], a 29 layer version of Resnet-34 [5] (used in the TrustID project) and iResnet-50 (with Squeeze and Excitation blocks)[6]. Both the iResnet-18 and iResnet-50 were trained on $112 \times 112 \times 3$ faces from the MS1MV2 dataset, using ArcFace as the loss function, and output 512-dimensional feature embeddings. Regarding the Resnet-34, it was trained with triplet loss on a custom dataset comprised of 3 million $150 \times 150 \times 3$ images from the Visual Geometry Group Face [93] and FaceScrub [90], and outputs a 128-dimensional feature embedding.

3.2.3 MobileFaceNet

MobileFaceNet [6] is the lightscale approach to face recognition that will further aid the study of the computational cost and model's performance trade-off. Being that is made available by the same author as the iResnet-50, it has technical similarities. Once again, it is trained with ArcFace loss on $112 \times 112 \times 3$ images from the MS1MV2 dataset and outputs the usual 512-dimensional embedding.

3.3 Finetuning data

Considering that the intended application of the face recognition systems is to monitor students, it is not unreasonable to assume that, due to the nature of the

[4] `https://github.com/deepinsight/insightface`

[5] `http://dlib.net/face_recognition.py.html`

[6] `https://github.com/TreB1eN/InsightFace_Pytorch`

capture device (webcam or smartphone) or its positioning, that will be reflect on the quality of the data. For instance, one potential scenario involves a student with one monitor and a laptop placed to the side of it, accompanied by a lamp on the opposite side. This will configuration will result in face captures with very different resolutions, poses and lightning compared to another student using a laptop, against a well lit window, facing the subject. Therefore, it is crucial to have a robust system that is capable of being insensitive to these variations, hence the possible need to finetune pretrained models for that. In this regard, we will examine two distinct datasets: DigiFace-1M and QMUL-SurvFace.

Amidst the controversies surrounding the methods of data acquisition for some publicly distributed datasets, including MS-Celeb-1M, MegaFace, FaceScrub, IJB-C or VGGFace, Bae *et al.* developed DigiFace-1M [6] with those concerns in mind. This fully synthetic dataset emulates different scenarios with variant poses, light, expression and accessories (hats, masks, makeup, etc.), and it serves as an interesting object of study of the potential of this type of datasets to diminish the reliance over real face data to finetune a model for more adverse scenarios.

The second dataset used to finetune our models of choice is QMUL-SurvFace [20] by Cheng *et al.*. Initially described as a benchmarking dataset, its unconstrained way of capturing resulted on faces with very high variance in resolution, capturing angles, poses, light or accessories, poses as a perfect source to further adapt the models to said scenarios. Another noteworthy benefit of this dataset is that the images we're collected with the consent of the individuals, meaning that ethical and privacy dilemmas are not at play.

3.4 Benchmarks

The evaluation will be conducted with 10-fold cross validation on the processed dataset to match the size of the training data for each model. First with the original pretrained model and then with the fine-tuned version of the selected one. The

model will produce the feature embeddings and pairwise cosine similarity, and if it is above a certain threshold the pair is considered as a match.

To comprehensively assess the performance of the chosen models, we will subject them to testing across eight diverse datasets: VGGFace2, AgeDB30, two CFP variations (CFP-FF for frontal-frontal pairs and CFP-FP for frontal-profile pairs), CALFW, CPLFW, XQLFW and LFW. This approach is designed to capture a wide array of characteristics, closely resembling real-world scenarios for a thorough evaluation, therefore, four dataset groups based on their characteristics are defined: frontal, age, pose and hard. The frontal group is composed of the CFP-FF and LFW, easier datasets with only frontal views of the face. The age group is designed to evaluate the model's sensitivity to age variations, and gathers the AgeDB30 and CALFW datasets. The pose one measures the system's performance when tested against faces with a wide range of poses by evaluating in the CFP-FP and CPLFW datasets. Finally, the hard group includes VGGFace2, a large scale benchmark with great variation in pose, age, light, etc., and XQLFW, a very difficult dataset with degraded image quality.

The chosen evaluation metrics are based on biometric systems of verification and encompass Accuracy, the Receiver Operating Characteristic (ROC) and Detection Error Trade-off (DET) plots obtained by sweeping an interval of thresholds and calculating the True Acceptance Rate (TAR), False Acceptance Rate (FAR) and the False Reject Rate (FRR), and the Equal Error Rate (EER). The $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$ is a measure for the number of correct predictions, $TAR = \frac{TP}{TP+FN}$ indicates the proportion of genuine face pairs correctly classified and $FAR = \frac{FP}{FP+TN}$ indicates the quantity of imposter face pairs incorrectly classified as matches, where TP is True Positive, TN is True Negative, FP is False Positive and FN is False Negative. With the previous metrics is possible to obtain the ROC plot and study the trade-off between the TAR and FAR at different thresholds and evaluate the performance of different models (Figure 18). $FRR = 1 - TAR$ calculates the ratio of genuine matches that are classified as a

non-match and allows to generate the DET curve which serves a similar purpose to the ROC curves, but with a better performance visualization due to the axis' logarithmic scale. Finally, the EER is the point where FRR = FAR and the lower it is, the better performing the system is.

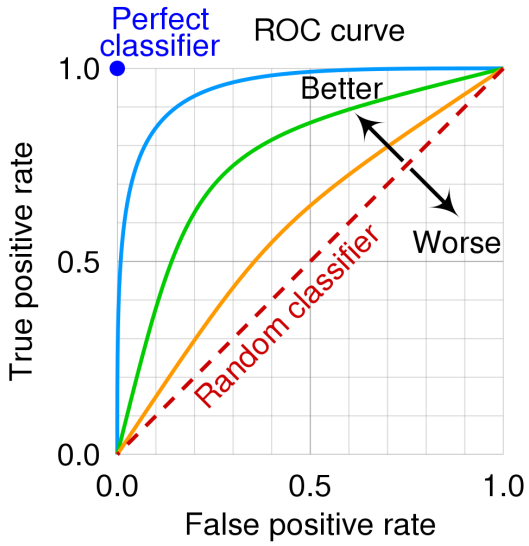


Figure 18: The y-axis represents the True Positive Rate (or TAR) and the x-axis represents the False Positive Rate (or FAR). The closer the ROC curve is to the top left corner, the better performance the model has, since that means that it is able of correctly identify more genuine face matches (TAR) while minimizing the incorrectly classified matches (FAR).

3.5 Implementation details

All the training, processing, and benchmarking code was developed in a Docker environment to ensure reproducibility. The code was written using Python 3.8.10 and relies on PyTorch 1.14.0 and Torchvision 0.15.0, along with their corresponding required libraries. The training process is conducted exclusively on a single GPU, specifically NVIDIA's GeForce RTX 3080 Ti 12Gb. Regarding the codes, they were developed with modularity in mind. First, the data is processed with the custom RetinaFace algorithm and saved. Then, it is proceeded forward to the

neural network that produces a feature embedding for each image. If the model is being trained, the embedding will be passed on to the ArcFace module, which will output the feature's logits. Following the original paper, the logits are turned to probabilities by applying the softmax activation function and then contributing to the final step, i.e, the cross entropy loss. This is handled by Pytorch's function CrossEntropyLoss, since it performs both steps simultaneously. Because ArcFace is a separate structure that does not integrate the backbone networks as a custom layer, the neural networks can be quickly and easily changed.

Chapter 4

Results

4.1 Models' specifications

We will compare the following four models against the TrustID Resnet-34 based solution, which comprises 29 layers, an average inference time of 7.28 seconds and has been trained on the VGG Face and Facescrub ensemble dataset using triplet loss. Relevant specifications that will assist in identifying the superior model are presented in Table 3. With Pytorch's model summary tool, the exact number of parameters, mult-adds and layers can be extracted, and from the models' documentation, the embedding size, loss used for training and the respective dataset.

	# Parameters	# Mult-Adds (G)	# Layers	Embedding	Inference time (s)	Loss	Dataset
MobileFaceNet	1,200,512	56.62	17	512	2.79	ArcFace	MS1MV2
iResnet-18	24,025,600	668.15	18	512	5.01	ArcFace	MS1MV2
FaceNet	28,907,599	152.21	63	512	5.89	Softmax	CASIA-WebFace
iResnet-SE-50	43,797,696	1610.00	50	512	9.78	ArcFace	MS1MV2

Table 3: Model's characteristics. "# Parameters" refers to the trainable parameters, "# Mult-Adds" to the number of multiplication and addition operations, "# Layers" denotes the quantity of convolutional and linear layers present in the model, "Embedding" signifies the dimensionality of the feature embedding produced by the model's output, "Inference time (s)" represents the average time taken for inference, "Loss" is the loss function used to train the model, and "Dataset" are the images employed for that action.

This initial analysis suggests that MobileFaceNet has promising characteristics. It is the least complex model, therefore, is less prone to overfitting to new data and, most importantly, the amount of computational overhead created and inferece times are much inferior to the others at study. This is supported by the fewer number of convolutional and linear layers, trainable parameters and mult-adds. Albeit the similar depth to that of iResnet-18, MobileFaceNet has, approximately, 95% fewer parameters, which is reflected on the number of total mult-adds. However, further investigation is required to determine whether the aforementioned characteristics might pose a bottleneck, potentially leading to a less robust solution with subpar performance. The ideal solution will strike a balance between adapting to new data and necessary computational costs.

4.2 Benchmarking Results

4.2.1 Accuracy

After performing 10-fold cross validation on all the benchmark datasets with the pretrained models, the mean accuracy is presented on the following table.

	Frontal		Age		Pose		Hard	
Models	CFP-FF	LFW	AgeDB30	CALFW	CFP-FP	CPLFW	VGGFace2	XQLFW
MobileFaceNet	0.9884	0.9912	0.9308	0.9362	0.8957	0.8642	0.9050	0.5063
iResnet-18	**0.9960**	**0.9960**	**0.9728**	**0.9555**	0.9414	0.8943	0.9198	0.4943
FaceNet	0.8909	0.9038	0.7147	0.7470	0.7664	0.6738	0.7748	0.5000
TrustID	0.8807	0.9906	0.7153	0.7198	0.7030	0.6235	0.7400	**0.6135**
iResnet-SE-50	0.9959	0.9953	0.9263	0.9543	**0.9457**	**0.9047**	**0.9396**	0.5137

Table 4: Model's face verification accuracy.

From Table 4, iResnet-18 achieves higher accuracy values on more datasets than any other model. However, iResnet-SE-50 performs better on the datasets where iResnet-18 falls short. Additionally, concerning the datasets where iResnet-SE-50 exhibited slightly lower performance, the accuracy scores are very close.

Specifically, CFP-FF, CFP-FP, CALFW, and LFW are within a margin of error that can be attributed to non-deterministic behaviors in PyTorch, the libraries used, hardware, and/or CUDA. It is also important to note that, even though MobileFaceNet didn't achieve the higher accuracy on any benchmark, the scores are the third best and considering its lightweight specifications highlighted in Table 3, the results are very promising and present a good example of accuracy and computational cost trade-off. Regarding the XQLFW results, with the exception of TrustID, they are exceedingly low, approaching 0.5 or worse. This suggests that the model is producing outputs that resemble random guesses, which is exactly what occurs with FaceNet.

4.2.2 ROC Curves

The previous table indicates that the three best performers, based exclusively on the accuracy at the best similarity threshold for each model and dataset, are the iResnet-SE-50, iResnet-18 and MobileFaceNet. To conduct a more thorough investigation allowing us to select the best model to be fine-tuned, the TAR values are calculated for a range of FAR values and the ROC curves are generated. By inspecting how close the ROC curve is to the top left corner, the best models can be determined since those are able to correctly identify more genuine matches while keeping the impostor matches low.

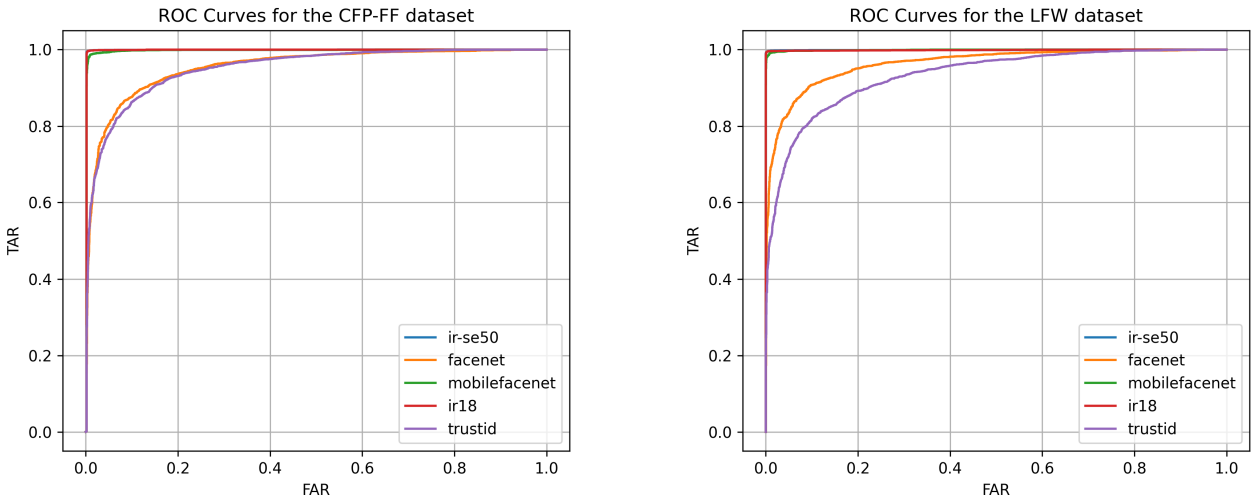


Figure 19: ROC Curves for the frontal pose face verification datasets.

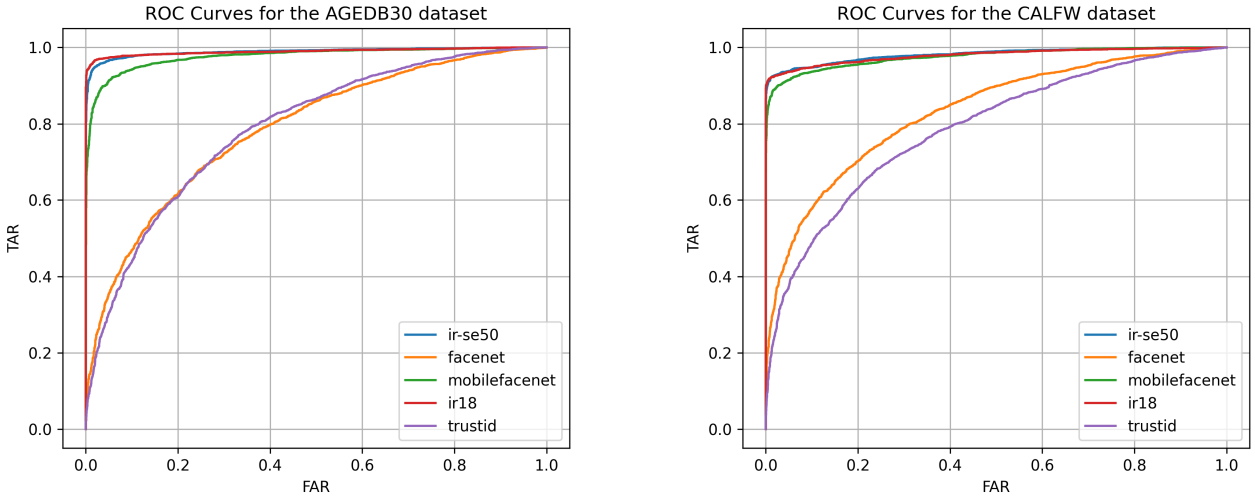


Figure 20: ROC Curves for the age face verification datasets.

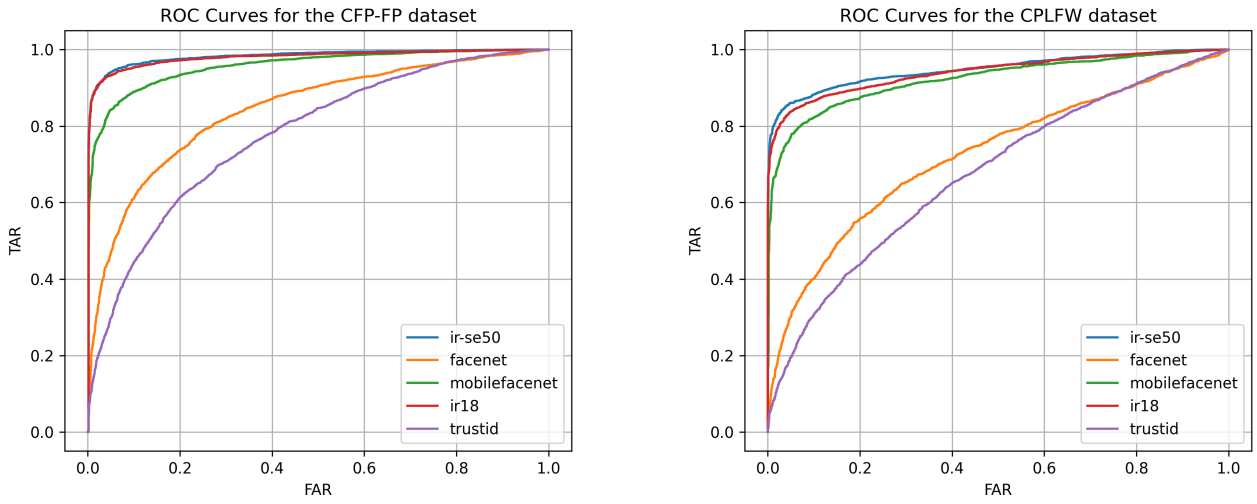


Figure 21: ROC Curves for the pose face verification datasets.

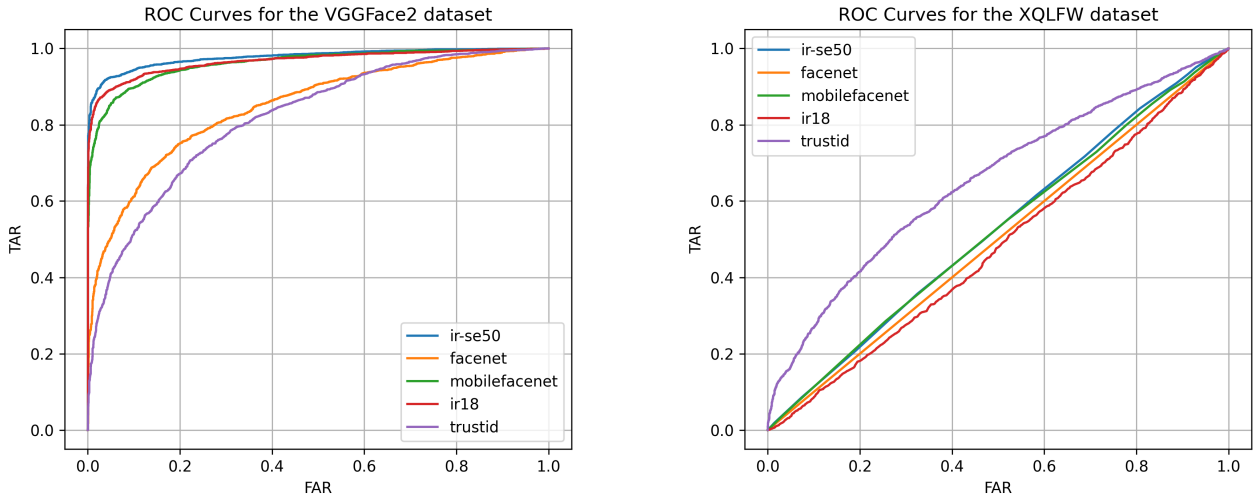


Figure 22: ROC Curves for the hard face verification datasets.

The ROC curves support our initial assumptions. It is evident that, across the entire FAR range, the three top-performing models are iResnet-SE-50, iResnet-18, and MobileFacenet. With the exception of XQLFW in (Figure 22), where all models, except TrustID, perform poorly and are close to random guessing, the

remaining seven datasets consistently position these three models near the top-left corner. This pattern indicates strong model performance. In scenarios with low FAR values, where the model is less tolerant of incorrectly identifying impostors as matches, the number of correctly classified pairs (TAR) is higher.

		iResnet-SE-50			iResnet-18		MobileFacenet			FaceNet		TrustID	
		$1e-4$	$1e-3$	1e-2	$1e-3$	$1e-2$	$1e-4$	$1e-3$	$1e-2$	$1e-3$	$1e-2$	$1e-3$	$1e-2$
Frontal	CFP-FF	0.10000	0.10000	0.99771	0.09886	0.99743	0.09057	0.09600	0.98657	0.02886	0.55257	0.03314	0.58114
	LFW	0.99067	0.99333	0.99700	0.99100	0.99600	0.91467	0.96933	0.99167	0.39900	0.68800	0.26967	0.50567
Age	AgeDB30	0.68267	0.81700	0.92600	0.91533	0.95300	0.4950	0.59500	0.80267	0.02900	0.14867	0.03467	0.11100
	CALFW	0.86633	0.88233	0.91733	0.90167	0.91933	0.68100	0.75900	0.87100	0.07867	0.26767	0.05300	0.18267
Pose	CFP-FP	0.07600	0.07971	0.87371	0.07628	0.87400	0.04200	0.05171	0.69857	0.00857	0.22171	0.00686	0.12629
	CPLFW	0.37533	0.58833	0.7900	0.66767	0.75400	0.06467	0.17200	0.64400	0.00967	0.12433	0.01567	0.07300
Hard	VGGFace2	0.06000	0.77280	0.86280	0.70320	0.81840	0.05240	0.53880	0.72160	0.17000	0.31720	0.06640	0.19400
	XQLFW	0.00001	0.00033	0.00800	0.00033	0.00433	0.00001	0.00100	0.00433	0.02000	0.40433	0.02167	0.07867

Table 5: TAR@FAR for all the models and benchmarks

The aforementioned three highest-achieving models distance themselves from FaceNet and TrustID on the ROC plots, although there is some overlap. Therefore, Table 5 allows us to analyze their performance at lower FAR values, where this overlap occurs.

For $FAR = 1e-4$, all models fail on more demanding datasets, but for easier ones iResnet-SE-50 is capable of good performance on LFW and CALFW, and MobileFaceNet on LFW. Reducing the strictness and increasing the FAR to $1e-3$, there's an improvement on the results as expected. The iResnet models produce high TAR values on all datasets apart from the more challenging CFP variations and XQLFW datasets, MobileFacenet starts to improve but still performs poorly on the pose group, hard group, CFP pair and AgeDB30 dataset. Finally, at $1e-2$ is the threshold at which all models excel without compromising the security of the system, since increasing the FAR to $1e-1$ would lead to too much falsely matched pairs. iResnet-SE-50 and iResnet-18 have comparable performance with high scores on the same benchmarks and both failing XQLFW. MobileFaceNet approaches iResnet levels of capability aside from slightly lower scores on the age

and pose groups and the usual XQLFW dataset.

4.2.3 DET Curves

To finalize the choice of the best model, the FRRs are calculated and plotted against the previous FAR values to obtain the DET curves. The intersection between the identity line that divides the graph and the DETs, the EER points can be extracted. These curves also allow to make a better distinction between models due to the more expansive logarithmic scale in which they are generated. In this case, contrary to the ROC curves, the better performing models are closer to the lower left corner, since that will minimize both the amount of impostors matched as true identities (FAR) and true identities classified as impostors (FRR).

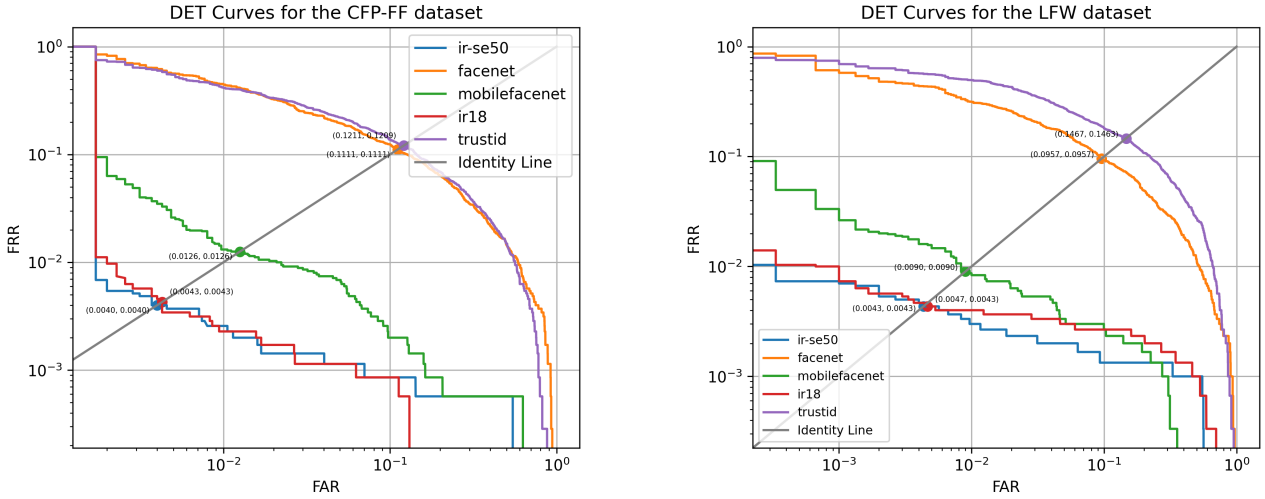


Figure 23: DTE Curves and EER points for the frontal pose face verification datasets.

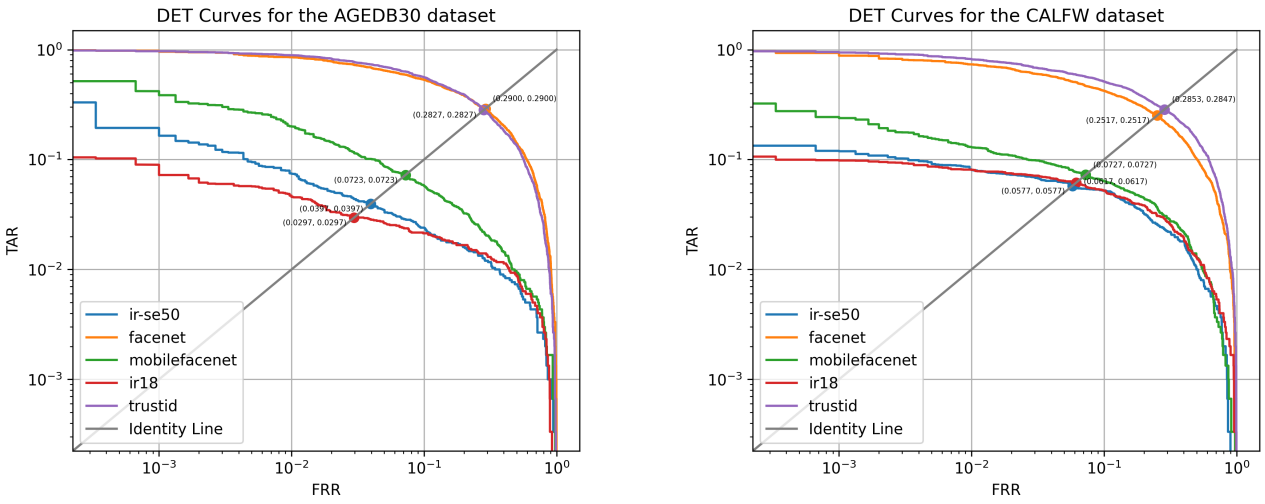


Figure 24: DET Curves and EER points for the age face verification datasets.

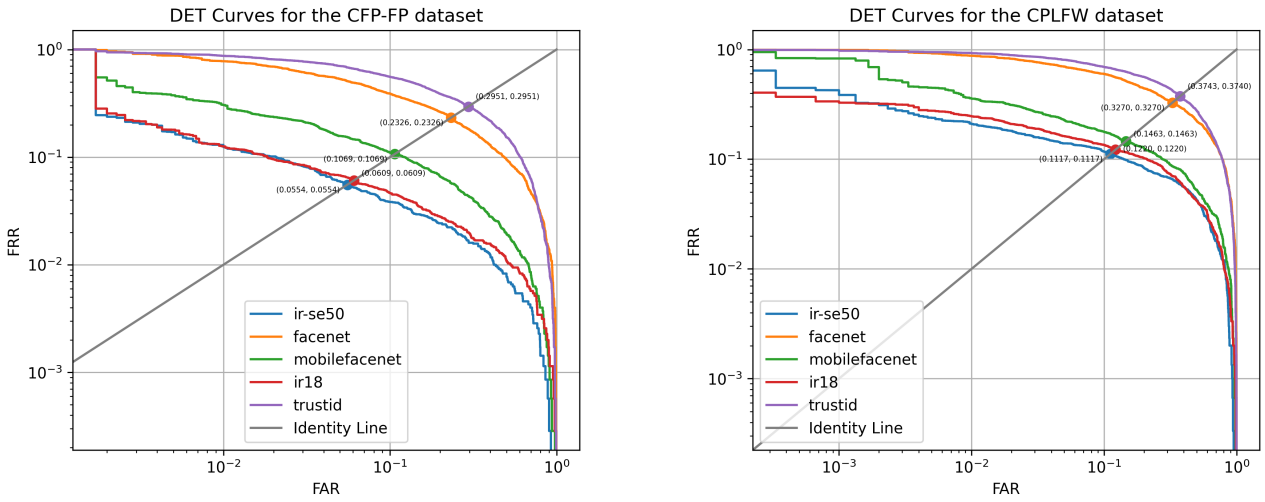


Figure 25: DET Curves and EER points for the pose face verification datasets.

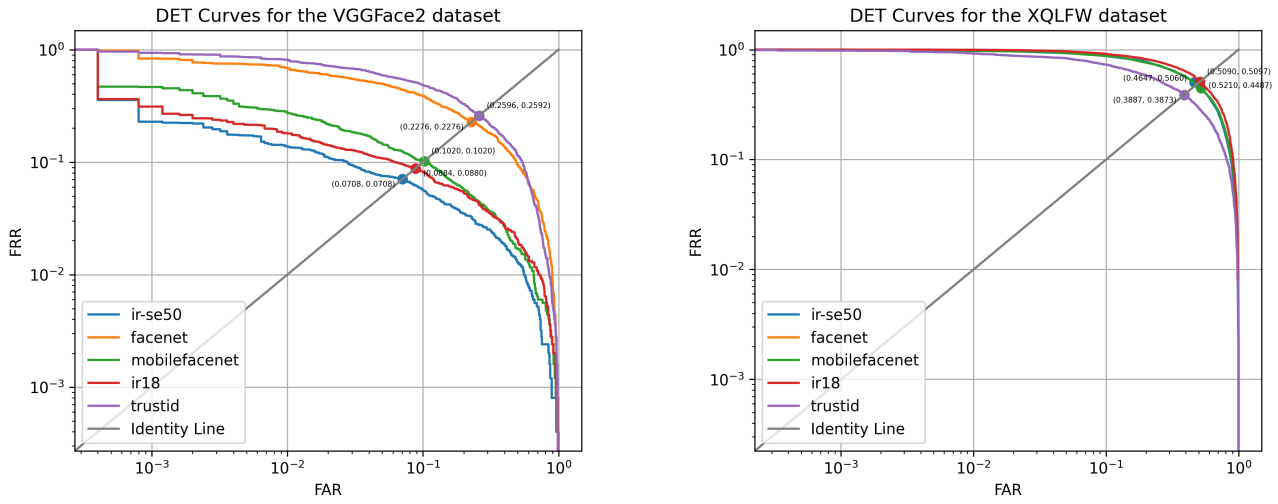


Figure 26: DET Curves and EER points for the hard face verification datasets.

	Frontal		Age		Pose		Hard	
	CFP-FF	LFW	AgeDB30	CALFW	CFP-FP	CPLFW	VGGFace2	XQLFW
MobileFaceNet	0.0126	0.0090	0.0723	0.0727	0.1069	0.1463	0.1020	0.4849
iResnet-18	0.0043	0.0045	**0.0297**	0.0617	0.0609	0.1220	0.0882	0.5094
FaceNet	0.1111	0.0957	0.2900	0.2517	0.2326	0.3270	0.2276	-
TrustID	0.1210	0.1465	0.2827	0.2850	0.2951	0.3742	0.2594	**0.3880**
iResnet-SE-50	**0.0040**	**0.0043**	0.0723	**0.0577**	**0.0554**	**0.1117**	**0.0708**	0.4854

Table 6: EER values for all the models and respective benchmarks.

As a final analysis, both the DET plots and the EER values support what has been previously discussed: iResnet-SE-50, iResnet-18 and MobileFaceNet are the best performing models. To no surprise, this group is always close to each other and close to the ideal corner of the DET graphs. Additionally, XQLFW reveals once again to be too much of a challenge. Regarding the EER scores, the results and conclusions are similar to the ones from Table 4. iResnet-SE-50, the bigger and more complex value, has the lower values, with iResnet-18 a close second and MobileFaceNet the third best.

4.2.4 Discussion

At this stage, we are capable of assuring, based on the previous tests, that there are better answers to TrustID's facial verification framework. iResnet-SE-50 is the overall best achieving model, iResnet-18 is the second best and MobileFaceNet is just slightly worse than the very close aforementioned two. The top three highlighted throughout prior sections proved their robustness to extreme variation in pose (CFP-FP and CPLFW), age (AgeDB30 and CALFW) or illumination (CFP-FF, VGGFace2). That being said, the benchmark concerning quality and image degradation (XQLFW) proved to be a major hurdle to most of the models aside from TrustID. That can be justified by the method of training. By resizing smaller training images to 150×150 there's a degradation in quality that leads to a model more prepared to handle these situations.

Considering the benchmark's results, the much lower amount of trainable parameters, mult-adds and inference time, MobileFaceNet is the best balance between performance and computational cost. Furthermore, although it is true that MobileFacenet's accuracy is high in the pose group and VGGFace2, there's room for improvement in the TAR at low FAR values, hence the fine-tuning. The objective of further training the model first with QMUL-SurvFace is to try and improve its scores in both the hard group, but specially XQLFW, and pose group benchmarks. On the other hand, DigiFace-1M enables the study of the impact of fully-synthetic ethically collected data on the scores throughout the benchmarks, with a special attention to the pose group benchmarks.

4.3 Training Details

By leveraging Optuna's hyperparameter searching capabilities, we concluded that the optimal combination is to train over batches of size 32 for 10 epochs with a $1e-4$ learning rate that decays according to a Cosine Annealing with warm up restarts. Additionally, because the model has Batch Normalization layers,

they need to be explicitly set to evaluation mode during training. If this step is overlooked, the mean and variance used will be the ones from the batch and not the values achieved during the pre-training, leading to bad evaluation values. Following the original ArcFace paper [27], the optimizer of choice is Stochastic Gradient Descent with momentum 0.9 and weight decay $5e-4$, scale $s = 64$ and margin $m = 0.5$. Early stopping, by evaluating on XQLFW and CPLFW during training, is also adopted as a safety measure to guarantee the best results possible.

4.4 Training Results

The first approach is to update the whole network, because both QMUL-SurvFace and DigiFace-1M are big datasets with a great number of images and identities, therefore, there are less chances of quickly overfitting to the training data.

Benchmarks		Original	$m = 0.5$	$m = 0.4$	$m = 0.3$
Frontal	CFP-FF	0.9884	0.7957 (↓)	0.7916 (↓)	0.7916 (↓)
	LFW	0.9912	0.7957 (↓)	0.7915 (↓)	0.7913 (↓)
Age	AgeDB30	0.9308	0.6165 (↓)	0.6112 (↓)	0.6362 (↓)
	CALFW	0.9362	0.6593 (↓)	0.6235 (↓)	0.6472 (↓)
Pose	CFP-FP	0.8957	0.6447 (↓)	0.6381 (↓)	0.6556 (↓)
	CPLFW	0.8642	0.6048 (↓)	0.5843 (↓)	0.5992 (↓)
Hard	VGGFace2	0.9050	0.6516 (↓)	0.6188 (↓)	0.6314 (↓)
	XQLFW	0.5063	0.5325 (↑)	0.5355 (↑)	0.5215 (↑)
Stopping Epoch			6	7	7

Table 7: MobileFaceNet accuracies before and after being fine-tuning the whole network on QMUL-SurvFace with different ArcFace margins.

According to Table 7, fine-tuning with QMUL-SurvFace improves, as would be

expected, the XQLFW benchmark performance by 5.17%. However, the verified increase is moderate and disadvantageous when the accuracy degradation verified on the remaining benchmarks is taken into account.

In an effort to improve the results, the margin is reduced in order to generate a less penalizing training with a more lax distance between classes. The results from this settings are also shown in Table 7, and when fine-tuned with QMUL-SurvFace, the XQLFW results are better than the original pre-trained model, increasing 5.77% for $m = 0.4$ and 3.00% for $m = 0.3$.

Benchmarks		Original	$m = 0.5$	$m = 0.4$	$m = 0.3$
Frontal	CFP-FF	0.9884	0.8011 (↓)	0.8231 (↓)	0.8157 (↓)
	LFW	0.9912	0.8011 (↓)	0.8231 (↓)	0.8157 (↓)
Age	AgeDB30	0.9308	0.6732 (↓)	0.6733 (↓)	0.6723 (↓)
	CALFW	0.9362	0.6755 (↓)	0.7010 (↓)	0.7010 (↓)
Pose	CFP-FP	0.8957	0.6191 (↓)	0.6483 (↓)	0.6609 (↓)
	CPLFW	0.8642	0.5945 (↓)	0.6078 (↓)	0.6170 (↓)
Hard	VGGFace2	0.9050	0.6520 (↓)	0.6744 (↓)	0.6758 (↓)
	XQLFW	0.5063	0.4965 (↓)	0.5003 (↓)	0.4975 (↓)
Stopping Epoch			6	5	6

Table 8: MobileFaceNet accuracies before and after being fine-tuning the whole network on DigiFace-1M with different ArcFace margins.

In the case of DigiFace-1M Table 8, no discernible improvements were observed, and reducing the margin size did not yield any positive changes. Instead, the model appeared to struggle in adapting to the dataset, resulting in adjustments to the weights that ultimately led to a deterioration in benchmark performance.

When comparing directly with the paper's suggested margin ($m = 0.5$), some conclusions can be drawn. The model fine-tuned with QMUL-SurvFace produces worse results in the frontal and age groups for $m = 0.4$ and $m = 0.3$, the pose

group is better for both margins and the hard group is mixed, where VGGFace2 has worse performance for $m = 0.4$ and $m = 0.3$, and XQLFW improves for $m = 0.4$ and not for $m = 0.3$. On the other hand, when $m = 0.4$ and $m = 0.3$, the DigiFace-1M training saw a marginal improvement throughout the tests but still performs poorly. All in all, reducing the margin size does not have a meaningful impact on the accuracy results.

To achieve a more profound understanding of how the models reacts to the data, Table 9 and Table 10 present the TAR at very low FAR.

		$m = 0.5$			$m = 0.4$			$m = 0.3$		
Benchmarks		$1e-4$	$1e-3$	$1e-2$	$1e-4$	$1e-3$	$1e-2$	$1e-4$	$1e-3$	$1e-2$
Frontal	CFP-FF	0.0140 (↓)	0.0209 (↓)	0.3486 (↓)	0.0063 (↓)	0.0154 (↓)	0.2940 (↓)	0.0091 (↓)	0.0169 (↓)	0.3097 (↓)
	LFW	0.3099 (↓)	0.3263 (↓)	0.4760 (↓)	0.1267 (↓)	0.1680 (↓)	0.3940 (↓)	0.1927 (↓)	0.2310 (↓)	0.4697 (↓)
Age	AgeDB30	0.0050 (↓)	0.0083 (↓)	0.0867 (↓)	0.0027 (↓)	0.0053 (↓)	0.0730 (↓)	0.0093 (↓)	0.0240 (↓)	0.0860 (↓)
	CALFW	0.0313 (↓)	0.8053 (↑)	0.8053 (↓)	0.0070 (↓)	0.0323 (↓)	0.0827 (↓)	0.0283 (↓)	0.0340 (↓)	0.0920 (↓)
Pose	CFP-FP	0.0006 (↓)	0.9380 (↑)	0.9380 (↑)	0.0003 (↓)	0.0003 (↓)	0.0666 (↓)	0.0000 (↓)	0.0003 (↓)	0.0626 (↓)
	CPLFW	0.0060 (↓)	0.9664 (↑)	0.9664 (↑)	0.0050 (↓)	0.0067 (↓)	0.0397 (↓)	0.0073 (↓)	0.0147 (↓)	0.0613 (↓)
Hard	VGGFace2	0.0040 (↓)	0.0536 (↓)	0.1060 (↓)	0.0036 (↓)	0.0376 (↓)	0.0880 (↓)	0.0040 (↓)	0.0364 (↓)	0.1216 (↓)
	XQLFW	0.0000 (↓)	0.0023 (↑)	0.0173 (↑)	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)

Table 9: TAR@FAR after fine-tuning the model with QMUL-SurvFace.

		$m = 0.5$			$m = 0.4$			$m = 0.3$		
Benchmarks		$1e-4$	$1e-3$	$1e-2$	$1e-4$	$1e-3$	$1e-2$	$1e-4$	$1e-3$	$1e-2$
Frontal	CFP-FF	0.0020 (↓)	0.0069 (↓)	0.1906 (↓)	0.0026 (↓)	0.0111 (↓)	0.3580 (↓)	0.0057 (↓)	0.0109 (↓)	0.3606 (↓)
	LFW	0.1633 (↓)	0.2810 (↓)	0.5520 (↓)	0.2300 (↓)	0.3403 (↓)	0.6310 (↓)	0.2320 (↓)	0.3743 (↓)	0.6113 (↓)
Age	AgeDB30	0.0013 (↓)	0.0083 (↑)	0.0530 (↓)	0.0010 (↓)	0.0153 (↓)	0.0747 (↓)	0.0003 (↓)	0.0250 (↓)	0.0790 (↓)
	CALFW	0.0017 (↓)	0.0043 (↓)	0.0780 (↓)	0.0013 (↓)	0.0300 (↓)	0.1207 (↓)	0.0050 (↓)	0.0170 (↓)	0.1223 (↓)
Pose	CFP-FP	0.0000 (↓)	0.0000 (↓)	0.0149 (↓)	0.0000 (↓)	0.0000 (↓)	0.0194 (↓)	0.0000 (↓)	0.0003 (↓)	0.0226 (↓)
	CPLFW	0.0013 (↓)	0.0017 (↓)	0.0190 (↓)	0.0007 (↓)	0.0016 (↓)	0.0263 (↓)	0.0007 (↓)	0.0033 (↓)	0.0287 (↓)
Hard	VGGFace2	0.0000 (↓)	0.0008 (↓)	0.0220 (↓)	0.0000 (↓)	0.0020 (↓)	0.0404 (↓)	0.0000 (↓)	0.0028 (↓)	0.0440 (↓)
	XQLFW	0.0000 (↓)	0.0000 (↓)	0.0083 (↓)	0.0003 (↑)	0.0013 (↑)	0.0087 (↑)	0.0000 (↓)	0.0010 (–)	0.0073 (↑)

Table 10: TAR@FAR after fine-tuning the model with DigiFace-1M.

In the context of general CNN architecture, it's well-established that the initial layers are primarily responsible for learning fundamental features such as edges, basic shapes, and patterns that constitute objects or faces. Therefore, with the

intention of improving the previous results, by preserving the weights associated with these earlier layers and avoiding introducing noise during further training, two other approaches are made: 1) freeze the first 5 layers and 2) train only the 2 final layers. Additionally, following the logic from the previous experience, different ArcFace margins are also studied.

4.4.1 5 layers

Benchmarks		Original	$m = 0.5$	$m = 0.4$	$m = 0.3$
Frontal	CFP-FF	0.9884	0.7821 (↓)	0.8414 (↓)	0.8155 (↓)
	LFW	0.9912	0.7820 (↓)	0.8415 (↓)	0.8156 (↓)
Age	AgeDB30	0.9308	0.5985 (↓)	0.6373 (↓)	0.6382 (↓)
	CALFW	0.9362	0.6506 (↓)	0.6962 (↓)	0.6765 (↓)
Pose	CFP-FP	0.8957	0.6477 (↓)	0.6627 (↓)	0.6499 (↓)
	CPLFW	0.8642	0.5943 (↓)	0.6195 (↓)	0.5872 (↓)
Hard	VGGFace2	0.9050	0.6442 (↓)	0.6822 (↓)	0.6610 (↓)
	XQLFW	0.5063	0.4925 (↓)	0.5020 (↓)	0.5127 (↑)
Stopping Epoch			8	5	6

Table 11: MobileFaceNet accuracies before and after being fine-tuning the network, with the first five layers frozen, on QMUL-SurvFace with different ArcFace margins.

Benchmarks		Original	$m = 0.5$	$m = 0.4$	$m = 0.3$
Frontal	CFP-FF	0.9884	0.8840 (↓)	0.8806 (↓)	0.8788 (↓)
	LFW	0.9912	0.8840 (↓)	0.8805 (↓)	0.8789 (↓)
Age	AgeDB30	0.9308	0.7265 (↓)	0.7125 (↓)	0.7303 (↓)
	CALFW	0.9362	0.7400 (↓)	0.7478 (↓)	0.7398 (↓)
Pose	CFP-FP	0.8957	0.7219 (↓)	0.7039 (↓)	0.7223 (↓)
	CPLFW	0.8642	0.6423 (↓)	0.6335 (↓)	0.6435 (↓)
Hard	VGGFace2	0.9050	0.7220 (↓)	0.7122 (↓)	0.7080 (↓)
	XQLFW	0.5063	0.4997 (↓)	0.4967 (↓)	0.5033 (↓)
Stopping Epoch			6	6	6

Table 12: MobileFaceNet accuracies before and after fine-tuning the network, with the first five layers frozen, on DigiFace-1M with different ArcFace margins.

		m=0.5			m=0.4			m=0.3		
Benchmarks		1e-4	1e-3	1e-2	1e-4	1e-3	1e-2	1e-4	1e-3	1e-2
Frontal	CFP-FF	0.0117 (↓)	0.0149 (↓)	0.2769 (↓)	0.0174 (↓)	0.0303 (↓)	0.4786 (↓)	0.0163 (↓)	0.0286 (↓)	0.3897 (↓)
	LFW	0.2033 (↓)	0.2277 (↓)	0.5047 (↓)	0.2907 (↓)	0.3473 (↓)	0.5870 (↓)	0.2597 (↓)	0.2743 (↓)	0.5533 (↓)
Age	AgeDB30	0.0137 (↓)	0.0207 (↓)	0.0597 (↓)	0.0087 (↓)	0.0180 (↓)	0.0643 (↓)	0.0063 (↓)	0.0177 (↓)	0.0777 (↓)
	CALFW	0.018 (↓)	0.0210 (↓)	0.0877 (↓)	0.0667 (↓)	0.0737 (↓)	0.1510 (↓)	0.0267 (↓)	0.0570 (↓)	0.1450 (↓)
Pose	CFP-FP	0.0000 (↓)	0.0009 (↓)	0.0617 (↓)	0.0011 (↓)	0.0029 (↓)	0.0871 (↓)	0.0009 (↓)	0.0020 (↓)	0.0694 (↓)
	CPLFW	0.0070 (↓)	0.0170 (↓)	0.0570 (↓)	0.0100 (↓)	0.0183 (↓)	0.0870 (↓)	0.0093 (↓)	0.0187 (↓)	0.0633 (↓)
Hard	VGGFace2	0.0032 (↓)	0.0352 (↓)	0.1112 (↓)	0.0056 (↓)	0.0712 (↓)	0.1516 (↓)	0.0040 (↓)	0.0528 (↓)	0.1304 (↓)
	XQLFW	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)	0.0000 (↓)	0.0040 (↓)	0.0000 (↓)	0.0003 (↓)	0.0037(↓)

Table 13: TAR@FAR after fine-tuning the model, with the first five layers frozen, on QMUL-SurvFace.

		m=0.5			m=0.4			m=0.3		
Benchmarks		1e-4	1e-3	1e-2	1e-4	1e-3	1e-2	1e-4	1e-3	1e-2
Frontal	CFP-FF	0.0189 (↓)	0.0354 (↓)	0.5574 (↓)	0.0183 (↓)	0.0303 (↓)	0.5200 (↓)	0.0294 (↓)	0.0346 (↓)	0.5539 (↓)
	LFW	0.3207 (↓)	0.5577 (↓)	0.7440 (↓)	0.2677 (↓)	0.5180 (↓)	0.7360 (↓)	0.4420 (↓)	0.5980 (↓)	0.7497 (↓)
Age	AgeDB30	0.0083 (↓)	0.0363 (↓)	0.1450 (↓)	0.0027 (↓)	0.0203 (↓)	0.1407 (↓)	0.0047 (↓)	0.0253 (↓)	0.1353 (↓)
	CALFW	0.0407 (↓)	0.0813 (↓)	0.2690 (↓)	0.0100 (↓)	0.0593 (↓)	0.2423 (↓)	0.0150 (↓)	0.0393 (↓)	0.2590 (↓)
Pose	CFP-FP	0.0006 (↓)	0.0026 (↓)	0.1231 (↓)	0.0003 (↓)	0.0029 (↓)	0.0880 (↓)	0.0003 (↓)	0.0017 (↓)	0.1106 (↓)
	CPLFW	0.0003 (↓)	0.0017 (↓)	0.0383 (↓)	0.0007 (↓)	0.0023 (↓)	0.0247 (↓)	0.0003 (↓)	0.0017 (↓)	0.0407 (↓)
Hard	VGGFace2	0.0000 (↓)	0.0176 (↓)	0.1732 (↓)	0.0000 (↓)	0.0056 (↓)	0.1392 (↓)	0.0000 (↓)	0.0040 (↓)	0.1468 (↓)
	XQLFW	0.0000 (↓)	0.0010 (–)	0.0100 (↑)	0.0003 (↑)	0.0007 (↓)	0.0067 (↑)	0.0000 (↓)	0.0010 (–)	0.0093 (↑)

Table 14: TAR@FAR after fine-tuning the model, with the first five layers frozen, on DigiFace-1M.

4.4.2 Final Layers

Benchmarks		Original	$m = 0.5$	$m = 0.4$	$m = 0.3$
Frontal	CFP-FF	0.9884	0.8751 (↓)	0.8821 (↓)	0.8786 (↓)
	LFW	0.9912	0.8751 (↓)	0.8821 (↓)	0.8785 (↓)
Age	AgeDB30	0.9308	0.7097 (↓)	0.7220 (↓)	0.7080 (↓)
	CALFW	0.9362	0.7762 (↓)	0.7833 (↓)	0.7752 (↓)
Pose	CFP-FP	0.8957	0.6729 (↓)	0.6714 (↓)	0.6719 (↓)
	CPLFW	0.8642	0.6585 (↓)	0.6635 (↓)	0.6635 (↓)
Hard	VGGFace2	0.9050	0.6992 (↓)	0.7014 (↓)	0.7006 (↓)
	XQLFW	0.5063	0.4975 (↓)	0.4993 (↓)	0.5010 (↓)
Stopping Epoch			6	3	6

Table 15: MobileFaceNet accuracies before and after fine-tuning the network, with all the layers frozen aside the last two, on QMUL-SurvFace with different ArcFace margins.

Benchmarks		Original	$m = 0.5$	$m = 0.4$	$m = 0.3$
Frontal	CFP-FF	0.9884	0.9568 (↓)	0.9630 (↓)	0.9574 (↓)
	LFW	0.9912	0.9569 (↓)	0.9629 (↓)	0.9574 (↓)
Age	AgeDB30	0.9308	0.8420 (↓)	0.8533 (↓)	0.8403 (↓)
	CALFW	0.9362	0.8672 (↓)	0.8795 (↓)	0.8643 (↓)
Pose	CFP-FP	0.8957	0.8133 (↓)	0.8199 (↓)	0.8171 (↓)
	CPLFW	0.8642	0.7650 (↓)	0.7830 (↓)	0.7697 (↓)
Hard	VGGFace2	0.9050	0.8158 (↓)	0.8290 (↓)	0.8134 (↓)
	XQLFW	0.5063	0.4923 (↓)	0.4995 (↓)	0.4993 (↓)
Stopping Epoch			5	3	4

Table 16: MobileFaceNet accuracies before and after fine-tuning the network, with all the layers frozen aside the last two, on DigiFace-1M with different ArcFace margins.

?? and **??** present the results for the aforementioned experiences. For both QMUL-SurvFace and DigiFace-1M, there are no improvements at any margin value and the XQLFW improvements are lost with the exception of $m = 0.3$. The only observable difference is the reduced deterioration of the original weights, and although it produces better results than the first experience where the whole network is trained, they are still worse than the pretrained model. A common pattern between the three training approaches is that, for $m = 0.4$, the model appears to reach its peak accuracy, and for both the partial trainings, that value is reached faster, resulting in less weight deterioration. That behavior is foreseen, since the model trains for less epochs, which leads the weights to be updated less times, preventing them from getting worse.

		m=0.5			m=0.4			m=0.3		
Benchmarks		1e-4	1e-3	1e-2	1e-4	1e-3	1e-2	1e-4	1e-3	1e-2
Frontal	CFP-FF	0.038 (↓)	0.0394 (↓)	0.6503 (↓)	0.0280 (↓)	0.0369 (↓)	0.6237 (↓)	0.0306 (↓)	0.0411 (↓)	0.6483 (↓)
	LFW	0.1793 (↓)	0.5243 (↓)	0.7050 (↓)	0.1727 (↓)	0.5347 (↓)	0.6920 (↓)	0.1863 (↓)	0.5210 (↓)	0.6910 (↓)
Age	AgeDB30	0.0210 (↓)	0.0283 (↓)	0.1320 (↓)	0.0373 (↓)	0.0390 (↓)	0.1457 (↓)	0.0213 (↓)	0.0327 (↓)	0.1243 (↓)
	CALFW	0.0477 (↓)	0.0977 (↓)	0.3130 (↓)	0.0653 (↓)	0.1137 (↓)	0.3323 (↓)	0.0500 (↓)	0.1073 (↓)	0.3047 (↓)
Pose	CFP-FP	0.0020 (↓)	0.0034 (↓)	0.1323 (↓)	0.0020 (↓)	0.0043 (↓)	0.1346 (↓)	0.0017 (↓)	0.0031 (↓)	0.1146 (↓)
	CPLFW	0.0170 (↓)	0.0453 (↓)	0.1567 (↓)	0.0207 (↓)	0.0503 (↓)	0.1607 (↓)	0.0213 (↓)	0.0457 (↓)	0.1683 (↓)
Hard	VGGFace2	0.0040 (↓)	0.0508 (↓)	0.2036 (↓)	0.0052 (↓)	0.0544 (↓)	0.2160 (↓)	0.0044 (↓)	0.0488 (↓)	0.1996 (↓)
	XQLFW	0.0003 (↑)	0.0003 (↓)	0.0080 (↓)	0.0003 (↓)	0.0007 (↓)	0.0110 (↑)	0.0003 (↓)	0.0013 (↑)	0.0123 (↑)

Table 17: TAR@FAR after fine-tuning the model, with all the layers frozen except the last two, on QMUL-SurvFace.

		m=0.5			m=0.4			m=0.3		
Benchmarks		1e-4	1e-3	1e-2	1e-4	1e-3	1e-2	1e-4	1e-3	1e-2
Frontal	CFP-FF	0.0734 (↓)	0.0794 (↓)	0.8874 (↓)	0.0743 (↓)	0.0817 (↓)	0.9245 (↓)	0.0734 (↓)	0.0794 (↓)	0.9009 (↓)
	LFW	0.8316 (↓)	0.8880 (↓)	0.9516 (↓)	0.8516 (↓)	0.8923 (↓)	0.9503 (↓)	0.8417 (↓)	0.8840 (↓)	0.9517 (↓)
Age	AgeDB30	0.1307 (↓)	0.1707 (↓)	0.4283 (↓)	0.2260 (↓)	0.2390 (↓)	0.4747 (↓)	0.1303 (↓)	0.1893 (↓)	0.4237 (↓)
	CALFW	0.3793 (↓)	0.4287 (↓)	0.6457 (↓)	0.4913 (↓)	0.5370 (↓)	0.6933 (↓)	0.4120 (↓)	0.4683 (↓)	0.6550 (↓)
Pose	CFP-FP	0.0137 (↓)	0.0300 (↓)	0.4074 (↓)	0.0194 (↓)	0.0251 (↓)	0.4469 (↓)	0.0120 (↓)	0.0217 (↓)	0.4206 (↓)
	CPLFW	0.1050 (↑)	0.2400 (↑)	0.3673 (↓)	0.0830 (↑)	0.2007 (↑)	0.3977 (↓)	0.0713 (↑)	0.1890 (↑)	0.3780 (↓)
Hard	VGGFace2	0.0192 (↓)	0.2220 (↓)	0.4572 (↓)	0.0192 (↓)	0.2620 (↓)	0.5116 (↓)	0.0152 (↓)	0.2328 (↓)	0.4632 (↓)
	XQLFW	0.0000 (↓)	0.0000 (↓)	0.0047 (↑)	0.0000 (↓)	0.0000 (↓)	0.0057 (↑)	0.0000 (↓)	0.0000 (↓)	0.0040 (↓)

Table 18: TAR@FAR after fine-tuning the model, with all the layers frozen except the last two, on DigiFace-1M.

Chapter 5

Conclusion

5.1 Conclusion

This work studied different neural networks models in order to propose a better approach to TrustID's facial verification module. The main objective was to find a good compromise between the maximum accurate performance and the minimum computational cost possible without compromising safety. To that extent, four different sized models were suggested: MobileFaceNet, FaceNet, iResnet-18 and iResnet-SE-50. First they were compared in terms of their specifications: number of trainable parameters, mult-adds, number of trainable layers, embedding size, inference time, loss function and training dataset. Then benchmarks were designed to test the methods in a wide range of possible scenarios.

5.2 Future work

The network does not have enough complexity to capture the datasets Investigate the effect of the digiface on a model trained from scratch and on benchmarks with facial accessories and makeup

Bibliography

[1] T. Ahonen, A. Hadid, and M. Pietikäinen. "Face Description with Local Binary Patterns: Application to Face Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.12 (2006). Cited By :4611, pp. 2037–2041. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2006.244 (cit. on p. 9).

[2] Arohan Ajit, Koustav Acharya, and Abhishek Samanta. "A Review of Convolutional Neural Networks". In: *2020 International Conference on Emerging Trends in Information Technology and Engineering (Ic-ETITE)*. Feb. 2020, pp. 1–5. DOI: 10.1109/ic-ETITE47903.2020.049 (cit. on pp. 11–13).

[3] Laith Alzubaidi et al. "Review of Deep Learning: Concepts, CNN Architectures, Challenges, Applications, Future Directions". In: *Journal of Big Data* 8.1 (Mar. 2021), p. 53. ISSN: 2196-1115. DOI: 10.1186/s40537-021-00444-8. (Visited on 02/09/2023) (cit. on pp. 10–12, 14, 15).

[4] Xiang An et al. *Partial FC: Training 10 Million Identities on a Single Machine*. Comment: 8 pages, 9 figures. Jan. 2021. arXiv: 2010.05222 [cs]. (Visited on 04/28/2023) (cit. on pp. 30, 31).

[5] Yousef Atoum et al. "Automated Online Exam Proctoring". In: *IEEE Transactions on Multimedia* 19.7 (July 2017), pp. 1609–1624. ISSN: 1941-0077. DOI: 10.1109/TMM.2017.2656064 (cit. on p. 49).

[6] Gwangbin Bae et al. “DigiFace-1M: 1 Million Digital Face Images for Face Recognition”. In: *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Datasets. Waikoloa, HI, USA: IEEE, Jan. 2023, pp. 3515–3524. ISBN: 978-1-66549-346-8. DOI: 10.1109/WACV56688.2023.00352. (Visited on 02/28/2023) (cit. on pp. 27, 30, 31, 56).

[7] Ankan Bansal et al. *The Do's and Don'ts for CNN-based Face Verification*. Comment: 10 pages including references, added more experiments on deeper vs wider dataset (section 3.2). Sept. 2017. arXiv: 1705.07426 [cs]. (Visited on 04/28/2023) (cit. on pp. 27, 28, 31).

[8] Maria Barron Rodriguez et al. *Remote Learning During the Global School Lockdown: Multi-Country Lessons*. World Bank, Aug. 2021. DOI: 10.1596/36141. (Visited on 03/13/2023) (cit. on p. 1).

[9] Chandrasekhar Bhagavatula et al. *Faster Than Real-time Facial Alignment: A 3D Spatial Transformer Network Approach in Unconstrained Poses*. Comment: International Conference on Computer Vision (ICCV) 2017. Sept. 2017. arXiv: 1707.05653 [cs]. (Visited on 04/15/2023) (cit. on p. 23).

[10] Fadi Boutros et al. “MixFaceNets: Extremely Efficient Face Recognition Networks”. In: *2021 IEEE International Joint Conference on Biometrics (IJCB)*. Aug. 2021, pp. 1–8. DOI: 10.1109/IJCB52358.2021.9484374 (cit. on pp. 37, 41).

[11] S. Charles Brubaker et al. “On the Design of Cascades of Boosted Ensembles for Face Detection”. In: *International Journal of Computer Vision* 77.1 (May 2008), pp. 65–86. ISSN: 1573-1405. DOI: 10.1007/s11263-007-0060-1 (cit. on p. 19).

[12] Jiajiong Cao, Yingming Li, and Zhongfei Zhang. “Celeb-500K: A Large Training Dataset for Face Recognition”. In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. Oct. 2018, pp. 2406–2410. DOI: 10.1109/ICIP.2018.8451704 (cit. on pp. 29, 31).

[13] Qiong Cao et al. *VGGFace2: A Dataset for Recognising Faces across Pose and Age*. Comment: This paper has been accepted by IEEE Conference on Automatic Face and Gesture Recognition (F&G), 2018. (Oral). May 2018. arXiv: 1710.08092 [cs]. (Visited on 04/27/2023) (cit. on pp. 27, 28, 31).

[14] Weipeng Cao et al. "A Review on Neural Networks with Random Weights". In: *Neurocomputing* 275 (Jan. 2018), pp. 278–287. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2017.08.040. (Visited on 02/09/2023) (cit. on p. 10).

[15] Fengju Chang et al. *FacePoseNet: Making a Case for Landmark-Free Face Alignment*. Aug. 2017. arXiv: 1708.07517 [cs]. (Visited on 04/15/2023) (cit. on pp. 23, 24).

[16] Kumar Chellapilla, Sidd Puri, and Patrice Simard. "High Performance Convolutional Neural Networks for Document Processing". In: () (cit. on p. 7).

[17] Lisha Chen, Hui Su, and Qiang Ji. "Face Alignment With Kernel Density Deep Neural Network". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 6991–7001. ISBN: 978-1-72814-803-8. DOI: 10.1109/ICCV.2019.00709. (Visited on 04/15/2023) (cit. on p. 23).

[18] Sheng Chen et al. *MobileFaceNets: Efficient CNNs for Accurate Real-Time Face Verification on Mobile Devices*. Comment: Accepted as a conference paper at CCBR 2018. Camera-ready version. June 2018. arXiv: 1804.07573 [cs]. (Visited on 02/27/2023) (cit. on pp. 37, 40).

[19] Zhiyi Cheng, Xiatian Zhu, and Shaogang Gong. "Low-Resolution Face Recognition". In: *Computer Vision – ACCV 2018*. Ed. by C. V. Jawahar et al. Vol. 11363. Cham: Springer International Publishing, 2019, pp. 605–621. ISBN: 978-3-030-20892-9 978-3-030-20893-6. DOI: 10.1007/978-3-030-20893-6_38. (Visited on 05/04/2023) (cit. on pp. 34, 36).

[20] Zhiyi Cheng, Xiatian Zhu, and Shaogang Gong. *Surveillance Face Recognition Challenge*. Comment: The QMUL-SurvFace challenge is publicly available at https://qmul-survface.github.io/. Aug. 2018. arXiv: 1804.09691 [cs]. (Visited on 04/28/2023) (cit. on pp. 35, 36, 56).

[21] Chengjun Liu and H. Wechsler. "Gabor Feature Based Classification Using the Enhanced Fisher Linear Discriminant Model for Face Recognition". In: *IEEE Transactions on Image Processing* 11.4 (Apr. 2002), pp. 467–476. ISSN: 1941-0042. DOI: 10.1109/TIP.2002.999679 (cit. on p. 9).

[22] D.C. Ciresan et al. "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images". In: *NIPS* 25 (2012). Export Date: 26 January 2023; Cited By: 92, pp. 2852–2860 (cit. on p. 8).

[23] Dan Cireşan et al. "A Committee of Neural Networks for Traffic Sign Classification". In: *The 2011 International Joint Conference on Neural Networks*. July 2011, pp. 1918–1921. DOI: 10.1109/IJCNN.2011.6033458 (cit. on p. 7).

[24] Dan C. Cireşan et al. "Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*. Ed. by Kensaku Mori et al. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, pp. 411–418. ISBN: 978-3-642-40763-5. DOI: 10.1007/978-3-642-40763-5_51 (cit. on p. 8).

[25] Dan Claudiu Cireşan et al. "Deep, Big, Simple Neural Nets for Handwritten Digit Recognition". In: *Neural Computation* 22.12 (Dec. 2010), pp. 3207–3220. ISSN: 0899-7667. DOI: 10.1162/NECO_a_00052. (Visited on 01/25/2023) (cit. on p. 7).

[26] T.F Cootes et al. "View-Based Active Appearance Models". In: *Image and Vision Computing* 20.9 (Aug. 2002), pp. 657–664. ISSN: 0262-8856. DOI: 10.1016/S0262-8856(02)00055-0 (cit. on p. 19).

[27] Jiankang Deng et al. “ArcFace: Additive Angular Margin Loss for Deep Face Recognition”. In: () (cit. on pp. 29, 31, 45, 46, 70).

[28] Jiankang Deng et al. *Joint Multi-view Face Alignment in the Wild*. Comment: submit to IEEE Transactions on Image Processing. Aug. 2017. arXiv: 1708.06023 [cs]. (Visited on 04/15/2023) (cit. on p. 23).

[29] Jiankang Deng et al. *RetinaFace: Single-stage Dense Face Localisation in the Wild*. May 2019. arXiv: 1905.00641 [cs]. (Visited on 04/13/2023) (cit. on pp. 19–22).

[30] Stuart E. Dreyfus. “The Computational Solution of Optimal Control Problems with Time Lag”. In: *IEEE Transactions on Automatic Control* 18.4 (1973). Cited by: 32, pp. 383–385. DOI: 10.1109/TAC.1973.1100330 (cit. on p. 7).

[31] Hang Du et al. “The Elements of End-to-end Deep Face Recognition: A Survey of Recent Advances”. In: *ACM Computing Surveys* 54.10s (Jan. 2022), pp. 1–42. ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3507902. (Visited on 03/07/2023) (cit. on pp. 18, 19, 21–24, 27, 31, 44, 47).

[32] Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. *Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark*. Comment: Accepted in Neurocomputing, Elsevier. June 2022. arXiv: 2109.14545 [cs]. (Visited on 07/26/2023) (cit. on p. 13).

[33] Ionut Cosmin Duta et al. “Improved Residual Networks for Image and Video Recognition”. In: *2020 25th International Conference on Pattern Recognition (ICPR)*. Milan, Italy: IEEE, Jan. 2021, pp. 9415–9422. ISBN: 978-1-72818-808-9. DOI: 10.1109/ICPR48806.2021.9412193. (Visited on 05/16/2023) (cit. on pp. 37, 40).

[34] D. Elizondo. “The Linear Separability Problem: Some Testing Methods”. In: *IEEE Transactions on Neural Networks* 17.2 (Mar. 2006), pp. 330–344.

ISSN: 1045-9227. DOI: 10.1109/TNN.2005.860871. (Visited on 01/24/2023) (cit. on p. 6).

[35] José Nuno Faria et al. "Image-Based Face Verification for Student Identity Management — the TRUSTID Case Study". In: *Adjunct Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization*. UMAP '23 Adjunct. New York, NY, USA: Association for Computing Machinery, June 2023, pp. 66–71. ISBN: 978-1-4503-9891-6. DOI: 10.1145/3563359.3597397. (Visited on 07/25/2023) (cit. on pp. 1, 50).

[36] B. Farley and W. Clark. "Simulation of Self-Organizing Systems by Digital Computer". In: *Transactions of the IRE Professional Group on Information Theory* 4.4 (1954), pp. 76–84. DOI: 10.1109/TIT.1954.1057468 (cit. on p. 5).

[37] Zhen-Hua Feng et al. *Wing Loss for Robust Facial Landmark Localisation with Convolutional Neural Networks*. Comment: 11 pages, 6 figures, 6 tables. Oct. 2018. arXiv: 1711.06753 [cs]. (Visited on 04/14/2023) (cit. on pp. 21, 23).

[38] Asep Hadian Sudrajat Ganidisastra and Yoanes Bandung. "An Incremental Training on Deep Learning Face Recognition for M-Learning Online Exam Proctoring". In: *2021 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*. Apr. 2021, pp. 213–219. DOI: 10.1109/APWiMob51111.2021.9435232 (cit. on pp. 49, 54).

[39] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearnin MIT Press, 2016 (cit. on pp. 15, 16, 18).

[40] Jiuxiang Gu et al. "Recent Advances in Convolutional Neural Networks". In: *Pattern Recognition* 77 (May 2018), pp. 354–377. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2017.10.013. (Visited on 02/09/2023) (cit. on pp. 11, 13, 14).

[41] Yandong Guo et al. *MS-Celeb-1M: A Dataset and Benchmark for Large-Scale Face Recognition*. July 2016. arXiv: 1607.08221 [cs]. (Visited on 04/25/2023) (cit. on pp. 28, 31).

[42] Michael Haenlein and Andreas Kaplan. "A Brief History of Artificial Intelligence: On the Past, Present, and Future of Artificial Intelligence". In: *California Management Review* 61 (July 2019), p. 000812561986492. DOI: 10.1177/0008125619864925 (cit. on p. 6).

[43] Munawar Hayat et al. "Joint Registration and Representation Learning for Unconstrained Face Identification". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI: IEEE, July 2017, pp. 1551–1560. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.169. (Visited on 04/15/2023) (cit. on p. 24).

[44] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 770–778. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.90. (Visited on 05/16/2023) (cit. on pp. 37, 39, 40).

[45] Kaiming He et al. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. Feb. 2015. arXiv: 1502.01852 [cs]. (Visited on 05/16/2023) (cit. on pp. 41, 42).

[46] Donald Olding Hebb. *The Organization of Behavior: A Neuropsychological Theory*. Wiley, 1949. ISBN: 978-0-471-36727-7 (cit. on p. 5).

[47] Sepp Hochreiter and Jürgen Schmidhuber. "Long Short-Term Memory". In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. (Visited on 01/25/2023) (cit. on p. 7).

[48] Seng Chun Hoo, Haidi Ibrahim, and Shahrel Azmin Suandi. "ConvFaceNeXt: Lightweight Networks for Face Recognition". In: *Mathematics* 10.19 (Jan.

2022), p. 3592. ISSN: 2227-7390. DOI: 10.3390/math10193592. (Visited on 05/16/2023) (cit. on pp. 37, 42).

[49] Jie Hu et al. *Squeeze-and-Excitation Networks*. Comment: journal version of the CVPR 2018 paper, accepted by TPAMI. May 2019. arXiv: 1709.01507 [cs]. (Visited on 05/16/2023) (cit. on p. 41).

[50] Peiyun Hu and Deva Ramanan. "Finding Tiny Faces". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI: IEEE, July 2017, pp. 1522–1530. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.166. (Visited on 04/14/2023) (cit. on p. 22).

[51] Gary B Huang et al. "Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments". In: () (cit. on pp. 31, 36, 47).

[52] Lichao Huang et al. *DenseBox: Unifying Landmark Localization with End to End Object Detection*. Sept. 2015. arXiv: 1509.04874 [cs]. (Visited on 04/13/2023) (cit. on p. 21).

[53] Xiehe Huang et al. *PropagationNet: Propagate Points to Curve to Learn Structure Information*. Comment: 10 pages, 8 figures, 8 tables, CVPR2020. June 2020. arXiv: 2006.14308 [cs]. (Visited on 04/08/2023) (cit. on p. 19).

[54] Yuge Huang et al. "CurricularFace: Adaptive Curriculum Learning Loss for Deep Face Recognition". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 5900–5909. ISBN: 978-1-72817-168-5. DOI: 10.1109/CVPR42600.2020.00594. (Visited on 02/28/2023) (cit. on p. 47).

[55] D. H. HUBEL and T. N. WIESEL. "Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex." In: *The Journal of physiology* 160.1 (Jan. 1962), pp. 106–154. ISSN: 0022-3751 1469-7793. DOI: 10.1113/jphysiol.1962.sp006837 (cit. on p. 12).

[56] A G Ivakhnenko and V G Lapa. “Cybernetic Predicting Devices”. In: () (cit. on p. 6).

[57] A. G. Ivakhnenko. “Polynomial Theory of Complex Systems”. In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-1.4 (1971), pp. 364–378. DOI: 10.1109/TSMC.1971.4308320 (cit. on p. 6).

[58] Roger David Joseph. *Contributions to Perceptron Theory*. Cornell Aeronautical Laboratory, 1960 (cit. on p. 6).

[59] Nathan D. Kalka et al. “IJB–S: IARPA Janus Surveillance Video Benchmark”. In: *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*. Oct. 2018, pp. 1–9. DOI: 10.1109/BTAS.2018.8698584 (cit. on pp. 9, 34, 36, 47).

[60] Xu Kang, Bin Song, and Fengyao Sun. “A Deep Similarity Metric Method Based on Incomplete Data for Traffic Anomaly Detection in IoT”. In: *Applied Sciences* 9 (Jan. 2019), p. 135. DOI: 10.3390/app9010135 (cit. on p. 10).

[61] Asifullah Khan et al. “A Survey of the Recent Architectures of Deep Convolutional Neural Networks”. In: *Artificial Intelligence Review* 53.8 (Dec. 2020), pp. 5455–5516. ISSN: 1573-7462. DOI: 10.1007/s10462-020-09825-6. (Visited on 02/09/2023) (cit. on p. 12).

[62] Minchul Kim, Anil K. Jain, and Xiaoming Liu. “AdaFace: Quality Adaptive Margin for Face Recognition”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. New Orleans, LA, USA: IEEE, June 2022, pp. 18729–18738. ISBN: 978-1-66546-946-3. DOI: 10.1109/CVPR52688.2022.01819. (Visited on 02/27/2023) (cit. on p. 47).

[63] Diederik P Kingma and Jimmy Lei. “Adam: A Method for Stochastic Optimization”. In: (2015) (cit. on p. 15).

[64] Brendan F. Klare et al. “Pushing the Frontiers of Unconstrained Face Detection and Recognition: IARPA Janus Benchmark A”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, June 2015, pp. 1931–1939. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298803. (Visited on 02/28/2023) (cit. on pp. 32, 36).

[65] Martin Knoche, Stefan Hormann, and Gerhard Rigoll. “Cross-Quality LFW: A Database for Analyzing Cross- Resolution Image Face Recognition in Unconstrained Environments”. In: *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*. Dec. 2021, pp. 1–5. DOI: 10.1109/FG52635.2021.9666960 (cit. on pp. 35, 36, 47).

[66] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc., 2012. (Visited on 01/26/2023) (cit. on pp. 8–10, 37).

[67] Mikel Labayen et al. “Online Student Authentication and Proctoring System Based on Multimodal Biometrics Technology”. In: *IEEE Access* 9 (2021), pp. 72398–72411. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3079375 (cit. on pp. 48, 50, 54).

[68] Erik Learned-Miller et al. “Labeled Faces in the Wild: A Survey”. In: *Advances in Face Detection and Facial Image Analysis*. Ed. by Michal Kawulok, M. Emre Celebi, and Bogdan Smolka. Cham: Springer International Publishing, 2016, pp. 189–248. ISBN: 978-3-319-25956-7 978-3-319-25958-1. DOI: 10.1007/978-3-319-25958-1_8. (Visited on 03/09/2023) (cit. on pp. 10, 19).

[69] Y. LeCun et al. “Backpropagation Applied to Handwritten Zip Code Recognition”. In: *Neural Computation* 1.4 (1989), pp. 541–551. DOI: 10.1162/neco.1989.1.4.541 (cit. on pp. 6, 16).

[70] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep Learning”. In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature14539 (cit. on pp. 10–13, 26).

[71] Yann LeCun et al. “Gradient-Based Learning Applied to Document Recognition”. In: (1998) (cit. on pp. 7, 9, 14, 37).

[72] Z. Lei, M. Pietikainen, and S.Z. Li. “Learning Discriminant Face Descriptor”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.2 (2014). Cited By :287, pp. 289–302. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2013.112 (cit. on p. 9).

[73] Stan Z. Li and Anil K. Jain, eds. *Handbook of Face Recognition*. London: Springer, 2011. ISBN: 978-0-85729-931-4 978-0-85729-932-1. DOI: 10.1007/978-0-85729-932-1. (Visited on 02/14/2023) (cit. on pp. 24, 25).

[74] Zewen Li et al. “A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects”. In: *IEEE Transactions on Neural Networks and Learning Systems* 33.12 (Dec. 2022), pp. 6999–7019. ISSN: 2162-2388. DOI: 10.1109/TNNLS.2021.3084827 (cit. on pp. 7, 8, 10, 12, 13).

[75] Seppo Linnainmaa. “The Representation of the Cumulative Rounding Error of an Algorithm as a Taylor Expansion of the Local Rounding Errors”. PhD thesis. Master’s Thesis (in Finnish), Univ. Helsinki, 1970 (cit. on p. 7).

[76] Hao Liu et al. “Two-Stream Transformer Networks for Video-Based Face Alignment”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.11 (Nov. 2018), pp. 2546–2554. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2017.2734779 (cit. on p. 23).

[77] Wei Liu et al. “SSD: Single Shot MultiBox Detector”. In: vol. 9905. Comment: ECCV 2016. 2016, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2. arXiv: 1512.02325 [cs]. (Visited on 04/13/2023) (cit. on p. 21).

[78] Weiyang Liu et al. *SphereFace: Deep Hypersphere Embedding for Face Recognition*. Comment: CVPR 2017 (v4: updated the Appendix). Jan. 2018. arXiv: 1704.08063 [cs]. (Visited on 02/28/2023) (cit. on pp. 45, 46).

[79] Yang Liu et al. *HAMBox: Delving into Online High-quality Anchors Mining for Detecting Outer Faces*. Comment: 9 pages, 6 figures. arXiv admin note: text overlap with 1802.09058 by other authors. Dec. 2019. arXiv: 1912.09231 [cs]. (Visited on 04/13/2023) (cit. on p. 21).

[80] Brais Martinez et al. "Local Evidence Aggregation for Regression-Based Facial Point Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.5 (May 2013), pp. 1149–1163. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2012.205 (cit. on p. 19).

[81] Brianna Maze et al. "IARPA Janus Benchmark - C: Face Dataset and Protocol". In: *2018 International Conference on Biometrics (ICB)*. Feb. 2018, pp. 158–165. DOI: 10.1109/ICB2018.2018.00033 (cit. on pp. 34, 36).

[82] J McCarthy et al. "A PROPOSAL FOR THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE". In: () (cit. on p. 5).

[83] Warren S Mcculloch and Walter Pitts. "A LOGICAL CALCULUS OF THE IDEAS IMMANENT IN NERVOUS ACTIVITY". In: () (cit. on p. 5).

[84] Qiang Meng et al. "MagFace: A Universal Representation for Face Recognition and Quality Assessment". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Nashville, TN, USA: IEEE, June 2021, pp. 14220–14229. ISBN: 978-1-66544-509-2. DOI: 10.1109/CVPR46437.2021.01400. (Visited on 02/28/2023) (cit. on p. 47).

[85] Shervin Minaee et al. *Going Deeper Into Face Detection: A Survey*. Mar. 2021 (cit. on p. 19).

[86] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA, USA: MIT Press, 1969 (cit. on p. 6).

[87] Stylianos Moschoglou et al. "AgeDB: The First Manually Collected, In-the-Wild Age Database". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Honolulu, HI, USA: IEEE, July 2017, pp. 1997–2005. ISBN: 978-1-5386-0733-6. DOI: 10.1109/CVPRW.2017.250. (Visited on 05/02/2023) (cit. on pp. 33, 36).

[88] Aaron Nech and Ira Kemelmacher-Shlizerman. "Level Playing Field for Million Scale Face Recognition". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Datasets. Honolulu, HI: IEEE, July 2017, pp. 3406–3415. ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.363. (Visited on 02/28/2023) (cit. on pp. 28, 31, 33, 36).

[89] Allen Newell, John C Shaw, and Herbert A Simon. "Report on a General Problem Solving Program". In: *IFIP Congress*. Vol. 256. Pittsburgh, PA. 1959, p. 64 (cit. on p. 5).

[90] Hong-Wei Ng and Stefan Winkler. "A Data-Driven Approach to Cleaning Large Face Datasets". In: *2014 IEEE International Conference on Image Processing (ICIP)*. Oct. 2014, pp. 343–347. DOI: 10.1109/ICIP.2014.7025068 (cit. on pp. 33, 50, 55).

[91] Kyoung-Su Oh and Keechul Jung. "GPU Implementation of Neural Networks". In: *Pattern Recognition* 37.6 (June 2004), pp. 1311–1314. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2004.01.013 (cit. on p. 7).

[92] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19.7 (July 1997), pp. 711–720. ISSN: 1939-3539. DOI: 10.1109/34.598228 (cit. on pp. 9, 19).

[93] Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman. "Deep Face Recognition". In: *Procedings of the British Machine Vision Conference 2015*. VGG-Face. Swansea: British Machine Vision Association, 2015, pp. 41.1–41.12. ISBN: 978-1-901725-53-7. DOI: 10.5244/C.29.41. (Visited on 02/27/2023) (cit. on pp. 16, 27, 28, 31, 50, 54, 55).

[94] B.T. Polyak. "Some Methods of Speeding up the Convergence of Iteration Methods". In: *USSR Computational Mathematics and Mathematical Physics* 4.5 (Jan. 1964), pp. 1–17. ISSN: 0041-5553. DOI: 10.1016/0041-5553(64)90137-5 (cit. on p. 15).

[95] Adrian Popescu et al. "Face Verification with Challenging Imposters and Diversified Demographics". In: *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Datasets. Waikoloa, HI, USA: IEEE, Jan. 2022, pp. 1151–1160. ISBN: 978-1-66540-915-5. DOI: 10.1109/WACV51458.2022.00122. (Visited on 02/28/2023) (cit. on pp. 35, 36).

[96] Rajat Raina, Anand Madhavan, and Andrew Y. Ng. "Large-Scale Deep Unsupervised Learning Using Graphics Processors". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. New York, NY, USA: Association for Computing Machinery, June 2009, pp. 873–880. ISBN: 978-1-60558-516-1. DOI: 10.1145/1553374.1553486. (Visited on 01/25/2023) (cit. on p. 7).

[97] Rajeev Ranjan, Carlos D. Castillo, and Rama Chellappa. *L2-Constrained Softmax Loss for Discriminative Face Verification*. L2-softmax Loss functions. June 2017. arXiv: 1703.09507 [cs]. (Visited on 02/28/2023) (cit. on p. 46).

[98] Rajeev Ranjan et al. "Deep Learning for Understanding Faces: Machines May Be Just as Good, or Better, than Humans". In: *IEEE Signal Processing Magazine* 35.1 (Jan. 2018), pp. 66–83. ISSN: 1558-0792. DOI: 10.1109/MSP.2017.2764116 (cit. on pp. 9, 18, 19, 22, 24, 26).

[99] Marc' aurelio Ranzato et al. "Efficient Learning of Sparse Representations with an Energy-Based Model". In: *Advances in Neural Information Processing Systems*. Vol. 19. MIT Press, 2006. (Visited on 01/25/2023) (cit. on p. 7).

[100] Shaoqing Ren et al. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. Comment: Extended tech report. Jan. 2016. arXiv: 1506.01497 [cs]. (Visited on 04/13/2023) (cit. on p. 20).

[101] F. Rosenblatt. "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain." In: *Psychological Review* 65 (1958), pp. 386–408. ISSN: 1939-1471(Electronic),0033-295X(Print). DOI: 10.1037/h0042519 (cit. on p. 5).

[102] Frank Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*. Spartan Books, 1962 (cit. on p. 6).

[103] Sebastian Ruder. "An Overview of Gradient Descent Optimization Algorithms". In: () (cit. on p. 14).

[104] DE Rumelhart, GE Hinton, and RJ Williams. *Learning Internal Representations by Error Propagation", in\Parallel Distributed Processing", DE Rumelhart, JL McClelland Eds*. 1986 (cit. on p. 7).

[105] Mark Sandler et al. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. Mar. 2019. arXiv: 1801.04381 [cs]. (Visited on 05/16/2023) (cit. on p. 40).

[106] Shreyak Sawhney et al. "Real-Time Smart Attendance System Using Face Recognition Techniques". In: *2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*. Jan. 2019, pp. 522–525. DOI: 10.1109/CONFLUENCE.2019.8776934 (cit. on p. 49).

[107] Jürgen Schmidhuber. “Deep Learning in Neural Networks: An Overview”. In: *Neural Networks* 61 (Jan. 2015), pp. 85–117. ISSN: 08936080. DOI: 10.1016/j.neunet.2014.09.003. (Visited on 01/24/2023) (cit. on p. 6).

[108] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “FaceNet: A Unified Embedding for Face Recognition and Clustering”. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682. arXiv: 1503.03832 [cs]. (Visited on 02/16/2023) (cit. on pp. 24, 31, 47, 54).

[109] Soumyadip Sengupta et al. “Frontal to Profile Face Verification in the Wild”. In: *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Celebrities in Frontal Profile (CFP)
Datasets. Mar. 2016, pp. 1–9. DOI: 10.1109/WACV.2016.7477558 (cit. on pp. 32, 36).

[110] P.Y. Simard, D. Steinkraus, and J.C. Platt. “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”. In: *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.* Vol. 1. Edinburgh, UK: IEEE Comput. Soc, 2003, pp. 958–963. ISBN: 978-0-7695-1960-9. DOI: 10.1109/ICDAR.2003.1227801. (Visited on 01/25/2023) (cit. on p. 7).

[111] Karen Simonyan and Andrew Zisserman. “VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION”. In: (2015) (cit. on pp. 37, 38).

[112] Irwin Sobel and Gary Feldman. “A 3×3 Isotropic Gradient Operator for Image Processing”. In: *Pattern Classification and Scene Analysis* (Jan. 1973), pp. 271–272 (cit. on p. 11).

[113] J. Stallkamp et al. “Man vs. Computer: Benchmarking Machine Learning Algorithms for Traffic Sign Recognition”. In: *Neural Networks*. Selected Pa-

pers from IJCNN 2011 32 (Aug. 2012), pp. 323–332. ISSN: 0893-6080. DOI: 10.1016/j.neunet.2012.02.016. (Visited on 01/25/2023) (cit. on p. 7).

[114] Yi Sun, Xiaogang Wang, and Xiaoou Tang. "Deep Learning Face Representation from Predicting 10,000 Classes". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. DeepID. June 2014, pp. 1891–1898. DOI: 10.1109/CVPR.2014.244 (cit. on p. 44).

[115] Christian Szegedy et al. *Going Deeper with Convolutions*. Sept. 2014. arXiv: 1409.4842 [cs]. (Visited on 05/16/2023) (cit. on pp. 37, 38).

[116] Yaniv Taigman et al. "DeepFace: Closing the Gap to Human-Level Performance in Face Verification". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, June 2014, pp. 1701–1708. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.220. (Visited on 02/13/2023) (cit. on pp. 9, 10).

[117] Yaniv Taigman et al. "Web-Scale Training for Face Identification". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA: IEEE, June 2015, pp. 2746–2754. ISBN: 978-1-4673-6964-0. DOI: 10.1109/CVPR.2015.7298891. (Visited on 02/28/2023) (cit. on p. 31).

[118] Mingxing Tan. "MixConv: Mixed Depthwise Convolutional Kernels". In: () (cit. on p. 41).

[119] Xu Tang et al. *PyramidBox: A Context-assisted Single Shot Face Detector*. Comment: 21 pages, 12 figures. Aug. 2018. arXiv: 1803.07737 [cs]. (Visited on 04/14/2023) (cit. on p. 22).

[120] Philipp Terhorst et al. "QMagFace: Simple and Accurate Quality-Aware Face Recognition". In: *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Waikoloa, HI, USA: IEEE, Jan. 2023, pp. 3473–3483. ISBN: 978-1-66549-346-8. DOI: 10.1109/WACV56688.2023.00348. (Visited on 02/28/2023) (cit. on p. 47).

[121] Jonathan Tompson et al. *Efficient Object Localization Using Convolutional Networks*. Comment: 8 pages with 1 page of citations. June 2015. arXiv: 1411.4280 [cs]. (Visited on 02/13/2023) (cit. on p. 10).

[122] A. M. Turing. "I.—COMPUTING MACHINERY AND INTELLIGENCE". In: *Mind* LIX.236 (Oct. 1950), pp. 433–460. ISSN: 1460-2113, 0026-4423. DOI: 10.1093/mind/LIX.236.433. (Visited on 01/13/2023) (cit. on p. 4).

[123] Matthew Turk and Alex Pentland. "Eigenfaces for Recognition". In: *Journal of Cognitive Neuroscience* 3.1 (Jan. 1991), pp. 71–86. ISSN: 0898-929X. DOI: 10.1162/jocn.1991.3.1.71. (Visited on 03/07/2023) (cit. on pp. 9, 19).

[124] Naeem Ullah et al. "A Novel DeepMaskNet Model for Face Mask Detection and Masked Facial Recognition". In: *Journal of King Saud University - Computer and Information Sciences* 34.10, Part B (Nov. 2022), pp. 9905–9914. ISSN: 1319-1578. DOI: 10.1016/j.jksuci.2021.12.017. (Visited on 02/27/2023) (cit. on pp. 35, 36).

[125] P. Viola and M. Jones. "Rapid Object Detection Using a Boosted Cascade of Simple Features". In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001.* Vol. 1. Dec. 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517 (cit. on p. 19).

[126] Fei Wang et al. *The Devil of Face Recognition Is in the Noise*. Comment: accepted to ECCV'18. July 2018. arXiv: 1807.11649 [cs]. (Visited on 04/28/2023) (cit. on pp. 29, 31).

[127] Feng Wang et al. "Additive Margin Softmax for Face Verification". In: *IEEE Signal Processing Letters* 25.7 (July 2018). AMS Loss, pp. 926–930. ISSN: 1070-9908, 1558-2361. DOI: 10.1109/LSP.2018.2822810. arXiv: 1801.05599 [cs]. (Visited on 02/28/2023) (cit. on p. 46).

[128] Feng Wang et al. "NormFace: L2 Hypersphere Embedding for Face Verification". In: *Proceedings of the 25th ACM International Conference on Multimedia*. Comment: camera-ready version. Oct. 2017, pp. 1041–1049.

DOI: 10.1145/3123266.3123359. arXiv: 1704.06369 [cs]. (Visited on 02/27/2023) (cit. on pp. 43–46).

[129] Hao Wang et al. *CosFace: Large Margin Cosine Loss for Deep Face Recognition*. Comment: Accepted by CVPR 2018. Apr. 2018. arXiv: 1801.09414 [cs]. (Visited on 02/28/2023) (cit. on p. 46).

[130] Mei Wang and Weihong Deng. "Deep Face Recognition: A Survey". In: *Neurocomputing* 429 (Mar. 2021), pp. 215–244. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2020.10.081. (Visited on 02/27/2023) (cit. on pp. 9, 18, 24, 26, 43).

[131] Mei Wang et al. "Racial Faces in the Wild: Reducing Racial Bias by Information Maximization Adaptation Network". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Korea (South): IEEE, Oct. 2019, pp. 692–702. ISBN: 978-1-72814-803-8. DOI: 10.1109/ICCV.2019.00078. (Visited on 05/05/2023) (cit. on pp. 34, 36).

[132] Nannan Wang et al. "Facial Feature Point Detection: A Comprehensive Survey". In: *Neurocomputing* 275 (Jan. 2018), pp. 50–65. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2017.05.013 (cit. on p. 19).

[133] Zhongyuan Wang et al. *Masked Face Recognition Dataset and Application*. Mar. 2020. arXiv: 2003.09093 [cs]. (Visited on 05/01/2023) (cit. on pp. 29, 31).

[134] Joseph Weizenbaum. "ELIZA—a Computer Program for the Study of Natural Language Communication between Man and Machine". In: *Communications of the ACM* 9.1 (Jan. 1966), pp. 36–45. ISSN: 0001-0782. DOI: 10.1145/365153.365168. (Visited on 01/18/2023) (cit. on p. 5).

[135] J. Weng, N. Ahuja, and T.S. Huang. "Cresceptron: A Self-Organizing Neural Network Which Grows Adaptively". In: *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*. Vol. 1. June 1992, 576–581 vol.1. DOI: 10.1109/IJCNN.1992.287150 (cit. on p. 7).

[136] Cameron Whitelam et al. “IARPA Janus Benchmark-B Face Dataset”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. July 2017, pp. 592–600. DOI: 10.1109/CVPRW.2017.87 (cit. on pp. 33, 36).

[137] Chris J Winstead. “Remote Microelectronics Laboratory Education in the COVID-19 Pandemic”. In: *2022 Intermountain Engineering, Technology and Computing (IETC)*. May 2022, pp. 1–6. DOI: 10.1109/IETC54973.2022.9796805 (cit. on p. 1).

[138] Lior Wolf, Tal Hassner, and Itay Maoz. “Face Recognition in Unconstrained Videos with Matched Background Similarity”. In: *CVPR 2011*. June 2011, pp. 529–534. DOI: 10.1109/CVPR.2011.5995566 (cit. on pp. 32, 36).

[139] Wayne Wu et al. *Look at Boundary: A Boundary-Aware Face Alignment Algorithm*. Comment: Accepted to CVPR 2018. Project page: https://wywu.github.io/projects/
May 2018. arXiv: 1805.10483 [cs]. (Visited on 04/15/2023) (cit. on p. 23).

[140] Shengtao Xiao et al. “Recurrent 3D-2D Dual Learning for Large-Pose Facial Landmark Detection”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. Venice: IEEE, Oct. 2017, pp. 1642–1651. ISBN: 978-1-5386-1032-9. DOI: 10.1109/ICCV.2017.181. (Visited on 04/15/2023) (cit. on p. 23).

[141] Yuanyuan Xu et al. *CenterFace: Joint Face Detection and Alignment Using Face as Point*. Comment: 11 pages, 3 figures. A demo of CenterFace can be available at https://github.com/Star-Clouds/CenterFace. Nov. 2019. arXiv: 1911.03599 [cs]. (Visited on 04/13/2023) (cit. on pp. 20–22).

[142] Rikiya Yamashita et al. “Convolutional Neural Networks: An Overview and Application in Radiology”. In: *Insights into Imaging* 9.4 (Aug. 2018), pp. 611–629. ISSN: 1869-4101. DOI: 10.1007/s13244-018-0639-9. (Visited on 02/09/2023) (cit. on pp. 10–12, 14).

[143] Mengjia Yan et al. *VarGFaceNet: An Efficient Variable Group Convolutional Neural Network for Lightweight Face Recognition*. Comment: 8 pages,2 figures. In Proceedings of the IEEE International Conference on Computer Vision Workshop, 2019. Nov. 2019. arXiv: 1910.04985 [cs]. (Visited on 02/27/2023) (cit. on pp. 37, 41).

[144] Ming Yang et al. "Detecting Human Actions in Surveillance Videos". In: 2009 TREC Video Retrieval Evaluation Notebook Papers. Cited by: 26. 2009 (cit. on p. 7).

[145] Dong Yi et al. *Learning Face Representation from Scratch*. Nov. 2014. arXiv: 1411.7923 [cs]. (Visited on 04/25/2023) (cit. on pp. 27, 31).

[146] Z. Cao et al. "Face Recognition with Learning-Based Descriptor". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. June 2010, pp. 2707–2714. ISBN: 1063-6919. DOI: 10.1109/CVPR.2010.5539992 (cit. on p. 9).

[147] Stefanos Zafeiriou, Cha Zhang, and Zhengyou Zhang. "A Survey on Face Detection in the Wild: Past, Present and Future". In: *Computer Vision and Image Understanding* 138 (Sept. 2015), pp. 1–24. ISSN: 1077-3142. DOI: 10.1016/j.cviu.2015.03.015 (cit. on p. 19).

[148] Andreas Zell. "Simulation Neuronaler Netze". In: 1994 (cit. on p. 10).

[149] Caiming Zhang and Yang Lu. "Study on Artificial Intelligence: The State of the Art and Future Prospects". In: *Journal of Industrial Information Integration* 23 (Sept. 2021), p. 100224. ISSN: 2452414X. DOI: 10.1016/j.jii.2021.100224. (Visited on 01/11/2023) (cit. on p. 5).

[150] Changzheng Zhang, Xiang Xu, and Dandan Tu. *Face Detection Using Improved Faster RCNN*. Feb. 2018. arXiv: 1802.02142 [cs]. (Visited on 04/13/2023) (cit. on p. 21).

[151] Yu-Dong Zhang et al. “Improved Breast Cancer Classification Through Combining Graph Convolutional Network and Convolutional Neural Network”. In: *Information Processing & Management* 58.2 (Mar. 2021), p. 102439. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2020.102439 (cit. on p. 10).

[152] Kaipeng Zhang et al. “Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks”. In: *IEEE Signal Processing Letters* 23.10 (Oct. 2016). Comment: Submitted to IEEE Signal Processing Letters, pp. 1499–1503. ISSN: 1070-9908, 1558-2361. DOI: 10.1109/LSP.2016.2603342. arXiv: 1604.02878 [cs]. (Visited on 04/13/2023) (cit. on pp. 19, 20, 22, 23).

[153] Qian Zhang et al. *VarGNet: Variable Group Convolutional Neural Network for Efficient Embedded Computing*. Comment: Technical report. Apr. 2020. arXiv: 1907.05653 [cs]. (Visited on 05/16/2023) (cit. on p. 41).

[154] Shifeng Zhang et al. *FaceBoxes: A CPU Real-time Face Detector with High Accuracy*. Comment: Accepted by IJCB 2017; Added references; Released codes. Dec. 2018. arXiv: 1708.05234 [cs]. (Visited on 04/14/2023) (cit. on p. 22).

[155] Zhou Zhang et al. “A Virtual Laboratory System with Biometric Authentication and Remote Proctoring Based on Facial Recognition”. In: *Computers in Education Journal* 7 (Dec. 2016), pp. 74–84 (cit. on p. 49).

[156] Zhou Zhang et al. “A Virtual Proctor with Biometric Authentication for Facilitating Distance Education”. In: *Lecture Notes in Networks and Systems*. Jan. 2018, pp. 110–124. ISBN: 978-3-319-64351-9. DOI: 10.1007/978-3-319-64352-6_11 (cit. on p. 49).

[157] He Zhao et al. “RDCFace: Radial Distortion Correction for Face Recognition”. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA: IEEE, June 2020, pp. 7718–7727.

ISBN: 978-1-72817-168-5. DOI: 10.1109/CVPR42600.2020.00774. (Visited on 02/28/2023) (cit. on p. 24).

[158] Jian Zhao, Shuicheng Yan, and Jiashi Feng. "Towards Age-Invariant Face Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.1 (Jan. 2022), pp. 474–487. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2020.3011426 (cit. on pp. 35, 36).

[159] Tianyue Zheng and Weihong Deng. "Cross-Pose LFW: A Database for Studying Cross-Pose Face Recognition in Unconstrained Environments". In: () (cit. on pp. 33, 35, 36).

[160] Tianyue Zheng, Weihong Deng, and Jiani Hu. *Cross-Age LFW: A Database for Studying Cross-Age Face Recognition in Unconstrained Environments*. Comment: 10 pages, 9 figures. Aug. 2017. arXiv: 1708.08197 [cs]. (Visited on 05/03/2023) (cit. on pp. 33, 35, 36).

[161] Xinqi Zhu and Michael Bain. *B-CNN: Branch Convolutional Neural Network for Hierarchical Classification*. Comment: 9 pages, 8 figures. Oct. 2017. DOI: 10.48550/arXiv.1709.09890. arXiv: 1709.09890 [cs]. (Visited on 02/13/2023) (cit. on p. 10).

[162] Zheng Zhu et al. "WebFace260M: A Benchmark Unveiling the Power of Million-Scale Deep Face Recognition". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Datasets. Nashville, TN, USA: IEEE, June 2021, pp. 10487–10497. ISBN: 978-1-66544-509-2. DOI: 10.1109/CVPR46437.2021.01035. (Visited on 02/28/2023) (cit. on pp. 30, 31).

[163] Fuzhen Zhuang et al. *A Comprehensive Survey on Transfer Learning*. Comment: 31 pages, 7 figures. June 2020. arXiv: 1911.02685 [cs, stat]. (Visited on 05/01/2023) (cit. on pp. 16, 36).