Imperial College London

Department of Computing

Advances in Generative Modelling: from Component Analysis to Generative Adversarial Networks

Stylianos Moschoglou

June 2021

Submitted in part fulfilment of the requirements for the degree of PhD in Computing and the Diploma of Imperial College London. This thesis is entirely my own work, and, except where otherwise indicated, describes my own research.

Abstract

This Thesis revolves around datasets and algorithms, with a focus on generative modelling. In particular, we first turn our attention to a novel, multi-attribute, 2D facial dataset. We then present deterministic as well as probabilistic Component Analysis (CA) techniques which can be applied to multi-attribute 2D as well as 3D data. We finally present deep learning generative approaches specially designed to manipulate 3D facial data.

Most 2D facial datasets that are available in the literature, are: a) automatically or semi-automatically collected and thus contain noisy labels, hindering the benchmarking and comparisons between algorithms. Moreover, they are not annotated for multiple attributes. In the first part of the Thesis, we present the first manually collected and annotated database, which contains labels for multiple attributes. As we demonstrate in a series of experiments, it can be used in a number of applications ranging from image translation to age-invariant face recognition.

Moving on, we turn our attention to CA methodologies. CA approaches, although being able to only capture linear relationships between data, can still be proven to be efficient in data such as UV maps or 3D data registered in a common template, since they are well aligned. The introduction of more complex datasets in the literature, which contain labels for multiple attributes, naturally brought the need for novel algorithms that can simultaneously handle multiple attributes. In this Thesis, we cover novel CA approaches which are specifically designed to be utilised in datasets annotated with respect to multiple attributes and can be used in a variety of tasks, such as 2D image denoising and translation, as well as 3D data generation and identification.

Nevertheless, while CA methods are indeed efficient when handling registered 3D facial data, linear 3D generative models lack details when it comes to reconstructing or generating finer facial characteristics. To alleviate this, in the final part of this Thesis we propose a novel generative framework harnessing the power of Generative Adversarial Networks.

Acknowledgements

When I first stepped foot in London, in October 2015, to join Imperial College London as a post-graduate Master's student, I would have never thought the time would come that I would be writing my Thesis. I first met Prof. Stefanos Zafeiriou when he was teaching the post-graduate, spring course "Advanced Statistical Machine Learning". From the first lecture he had me. I was so captivated by his passion and astounding depth in statistical learning that my decision to then do a PhD under his supervision only came naturally. I feel incredibly lucky to have met him and I sincerely thank him not only for his devoted academic support throughout my studies, but also for being there for me as a good friend, his advice, the endless conversations about politics and history and all the laughs as well as hardships we have been through that forged a unique and strong relationship.

Moreover, I would like to thank Dr. Mihalis Nicolaou for his guidance and academic support during these amazing years. I think that now is the right time to apologise for calling him at midnight to make a final proof-reading of my papers, only an hour (or was it merely a handful of minutes?) before the submission deadline. Mihali, at least we can now be both proud and say that it was not for nothing; the papers got accepted!

I would also like to express my gratitude and respect to Dr. Yannis Panagakis, who spent hours with me in the lab, sometimes even till very late in the evening until the security guard came to kick us out (!). Many of the proofs and derivations you will read in this Thesis would

have not been possible without his support.

Furthermore, I would like to deeply thank my colleagues in the i-BUG group for all the interesting and fruitful conversations we had. In particular, I would like to thank Griroris, Vangelis, Giorgos, Thanos, Alexandros, Jiankang, Baris, and Panagiotis for all the insightful talks and collaborations. However, first and foremost, I would like to thank my dear friend Stelios Ploumpis for not only having the most amazing and effortless academic collaboration with, but also for being a real brother to me in all (even the extraordinarily difficult) circumstances.

I would also like to thank Imperial College London for offering me the EPSRC Scholarship and funding my PhD studies as well as Mrs. Nathene Arnaoutis for her continuous support all these years. A huge thank you goes also to my girlfriend for always being there for me.

My deepest gratitude and appreciation goes to my parents who have fervently supported me throughout my whole life. This Thesis is dedicated to them.

Contents

List of Figures

List of Algorithms

List of Tables

Abbreviations

3DMM	3D Morphable Model
AE	AutoEncoder
ANN	Artificial Neural Network
BEGAN	Boundary Equilibrium Generative Adversarial Network
CA	Component Analysis
CCA	Canonical Correlation Analysis
CGAN	conditional Generative Adversarial Network
CoMA	Convolutional Mesh Autoencoder
CONV	Convolution
DCGAN	Deep Convolutional Generative Adversarial Network
DECONV	Deconvolution
DL	Deep Learning
DS-LDA	Dual Space Linear Discriminant Analysis
ELU	Exponential Linear Unit
ET	Entry-wise soft Thresholding
GAN	Generative Adversarial Network
JS	Jensen-Shannon divergence
KL	Kullback-Leibler divergence

LDA	Linear Discriminant Analysis
MAPLDA	Multi-Attribute Probabilistic Linear Discriminant Analysis
MARCA	Multi-Attribute Robust Component Analysis
OMRPCA	Optimal Mean Robust Principal Component Analysis
PCA	Principal Component Analysis
PGAN	Progressive Generative Adversarial Network
PLDA	Probabilistic Linear Discriminant Analysis
RCICA	Robust Correlated and Individual Component Analysis
RELU	Rectified Linear Unit
RJIVE	Robust Joint and Individual Variance Explained
RPCA	robust Principal Component Analysis
RPCA-L1	Robust Principal Component Analysis with Non-Greedy $l1$-Norm Maximisation
SVD	Singular Value Decomposition
SVT	Singular Value Thresholding
VAE	Variational AutoEncoder
WGAN-GP	Wasserstein Generative Adversarial Network with Gradient Penalty
WGAN	Wasserstein Generative Adversarial Network

Notation

$\sum$	Summation
$\prod$	Product
$\mathbb{R}$	Field of Real numbers
$\lvert x \rvert$	absolute value of x
$\sqrt{x}$	square root of x
$p(x)$	Probability Density Function (PDF) of x
$p(x\lvert y)$	Conditional PDF of x given y
$\mathcal{N}(\mu, \sigma^2)$	normal distribution with mean μ and variance σ^2
$\mathbf{x}$	column-vector with components x_i
$\mathbf{X}$	matrix with components X_{ij}
$\mathbf{X}^{-1}$	inverse of $\mathbf{X}$
$\mathbf{X}^{\mathsf{T}}$	transpose
$\mathbb{E}[x]$	Expectation of x
$\mathbf{S}$	Sample Covariance Matrix
$\mathbf{I}_k$	Identity $k \times k$-matrix
$\mathbf{1}_n$	is a vector of n ones
∇_x	vector differential operator over x
∂x	partial derivative of x

$\|\mathbf{X}\|_F$	Frobenius norm of $\mathbf{X}$
$\mathbf{X} \bigodot \mathbf{Y}$	Hadamard product of $\mathbf{X}$ and $\mathbf{Y}$
$\mathbf{X} \otimes \mathbf{Y}$	Kronecker product of $\mathbf{X}$ and $\mathbf{Y}$

Chapter 1

Introduction

Contents

1.1 Problem Scope

Since the dawn of civilised society, mankind has been immensely interested in figuring out how things work. In this attempt, analytical thinking was developed and evolved throughout the centuries. One main characteristic trait of analytical thinking that we use in order to understand things is the process of splitting wholes in their essential parts, which is a sort of "divide and conquer" approach. For instance, atomic energy was only discovered when we were able to delve into the atom level and subsequently learn more about the particles it is comprised of.

In a similar fashion, the majority of algorithms that have been utilised in the Computer Vision field throughout the past decades, try to precisely decompose data in their essential parts and then use these parts to conduct inferences about them. For example, when we are given an image depicting a subject and we are trying to infer their age, we would ideally like to extract the age component/part of the image which would give us the most meaningful information about this task. This set of algorithms that decompose data into their distinct parts or *components* comprises an area in the Computer Vision community that is called *Component Analysis* (CA). Over the past decades, numerous CA algorithms have been developed and utilised in a plethora of tasks such as 2D/3D face analysis. For instance, some of the most notable works on face identification [1, 2, 3], facial expression recognition [4, 5, 6], and image imputation [7, 8, 9], are based on CA.

Although CA methodologies can only capture the linear relationships between data, they can still be proven to be very efficient in generative tasks when handling data which are well-aligned, such as 2D UV maps [10] or 3D data registered under a common reference template [11, 12]. As a result, despite the progress of Deep Learning (DL) methodologies tailored to capturing higher-order relationships between data, linear CA models are still prevalent in the literature and provide state-of-the-art results in face analysis tasks such as 3D face reconstruction [12].

Nevertheless, as years went by, Computer Vision tasks became more challenging as more complex datasets appeared in the literature, providing data annotated with respect to *multiple labels as well as multiple attributes* (e.g., datasets annotated with respect to attributes like age and hair colour and various labels depending on the age-group and hair colour, respectively) [13, 14, 15]. One major drawback of most such multi-attribute datasets is that the labels they come with are automatically or semi-automatically assigned and are thus noisy. As a result, the algorithms that are evaluated using such datasets are prone to error. In the first part of this Thesis, we present the first manually collected and annotated multi-attribute dataset

[16]. We also utilise it in the Chapters that follow to demonstrate the merits of our novel CA methodologies. In Fig. 1.1 we showcase what the term "multi-attribute" stands for.

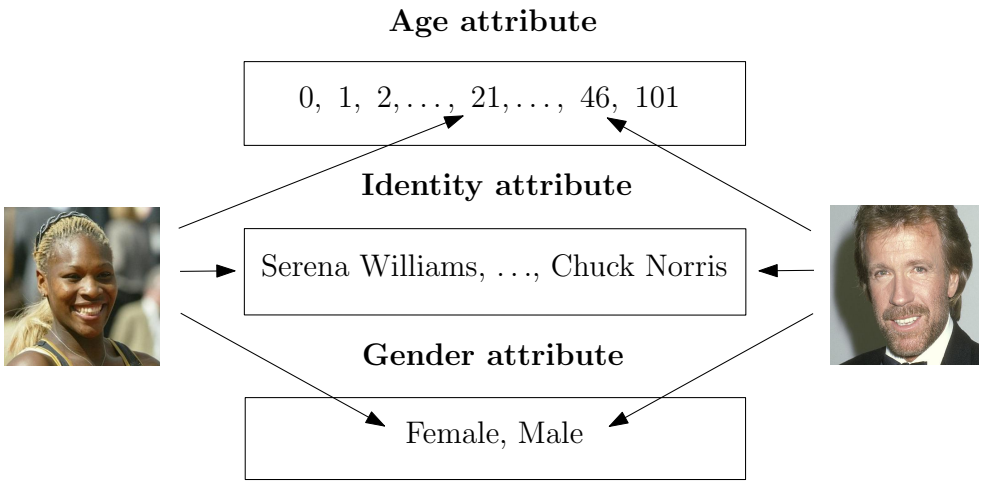


Figure 1.1: AgeDB is a dataset that includes annotations for multiple attributes (i.e., age, identity, and gender, as shown above). Moreover, each attribute is split into different classes/labels. For example, the attribute identity includes a number of different identities, such as Serena Williams. Each image in this multi-attribute dataset is annotated with a label/class pertaining to each attribute. We use the term "multi-attribute" to distinguish it from other datasets that may be multi-class/multi-label but are only annotated with respect to one attribute.

This introduction of more complex and multi-attribute datasets in the literature, naturally necessitated the need to extend widely used CA approaches so that they can handle multi-attribute data. In this Thesis, we detail a few new CA methodologies which are specifically tailored to multi-attribute 2D/3D data and provide state-of-the-art results in tasks including 3D face recognition [11] and 2D image denoising and translation [10].

While CA methodologies have been proven to be performing extremely satisfactorily in mainly classification tasks when applied on well aligned data, the advent of DL has brought about a tremendous change in the world of Computer Vision [17, 18, 19, 20, 21]. The unique capability of DL methodologies to capture higher-order relationships between data has made them the "gold standard" especially in generative tasks, where CA methods fall short of due to their linearity limitation discussed above. For example, DL approaches are the most prevalent

in the task of 2D image generation [22, 23, 24, 25, 26].

Nonetheless, even though generative DL approaches are ubiquitous in 2D data manipulation, only in the last couple of years have we begun to see works in the literature targeted to 3D manifolds such as meshes [27, 28, 29, 30]. This is mainly attributed to the fact that 2D data are Euclidean and thus operators such as the 2D convolution can be applied. On the other hand, 3D meshes are non-Euclidean and hence it is non-trivial to apply DL methods on such data. In the penultimate Chapter of this Thesis, we present some novel generative DL approaches with a focus on 3D facial data generation and showcase how we can circumvent the aforementioned obstacles and eventually achieve state-of-the-art results.

1.2 Objectives

In this Thesis, we present a new multi-attribute dataset and explore novel algorithms in two different areas of Machine Learning (ML), namely CA and DL. Having firstly presented our novel, manually collected, multi-attribute dataset, we then report novel deterministic as well as probabilistic CA methods which can be applied in multi-attribute data, including our dataset. Finally, in the last part, we present novel DL algorithms tailored to 3D facial data which can be used in generative tasks. In particular, we try to tackle the following problems:

1. In the last years, more and more challenging datasets have been introduced in the literature, containing labels for multiple attributes. Nevertheless, such datasets have inherent drawbacks as they have been annotated by either semi-automatic or completely automatic ways. Therefore, they have noisy labels and hence they are prone to noisy estimates when they are used as benchmarks to evaluate the performance of algorithms. As a result, could we introduce a new multi-attribute dataset in the literature that is manually collected, annotated, and curated in order to create a leading standard when it comes to evaluating the performance of state-of-the-art algorithms tailored to face

analysis applications?

2. Recently, more and more complex 2D/3D facial datasets appear in the literature annotated with respect to multiple attributes. Although there exist many CA algorithms in the literature that can be used in data annotated with respect to only one attribute, there are not many approaches that can be utilised under the multi-attribute scenario by simultaneously exploiting all the available information with respect to the various attributes. As a result, could we create novel CA algorithms that exploit all of the attribute information at the same time and thus improve the performance in a variety of tasks, including but not limited to 3D face identification or 2D facial image denoising?

3. DL methodologies have revolutionised the field of Computer Vision especially in 2D generative tasks, by exploiting the power of Generative Adversarial Networks (GANs). However, even though GANs are widely used in 2D data, GAN approaches in 3D manifolds are still in their infancy mainly because the 3D data are non-Euclidean and hence very useful operators such as the 2D convolution cannot be used. As a result, could we introduce novel GAN approaches that are able to work well with complex 3D meshes?

Objective 1 Based on the plethora of publicly available facial images of celebrities on-line, we can manually collect a large pool of such data. Moreover, based on the meta-info of the images, we can label them according to the identity of the celebrity, gender, and age (which can be accurate to the year). By doing so, we will have manually collected and annotated a dataset which carries labels of multiple attributes such as identity, gender, and age. We can then benchmark this dataset on a variety of tasks such as age-invariant face recognition, age estimation and age progression "in-the-wild" utilising state-of-the-art algorithms and thus make it the standard benchmark for these tasks.

Objective 2 Most widely used CA approaches in the literature work under the single attribute assumption. We can extend them so that they can be used in multiple attribute

scenarios. In particular, we turn our attention to deterministic as well as probabilistic formulations of known CA methods. We apply the said methods in well-aligned data such as 2D facial UV maps or 3D facial data registered in a common reference template and showcase their superiority against their counterparts in classification as well as generation tasks.

Objective 3 Even though 2D convolutions cannot be directly applied on 3D data due to their non-Euclidean structure, if we managed to transfer the 3D data in the 2D space, we could still apply the 2D convolutions on them. A computationally cheap and efficient way to do that on registered 3D facial data is to use a fixed UV map topology and then apply this pre-determined 3D to 2D transformation to all data. Having achieved this, we can harness the power of GANs (including the usage of the convolution operator) and create novel GAN losses and architectures tailored to such data. As we demonstrate, such GAN frameworks outperform approaches based on geometric DL.

1.3 Contributions — Thesis Overview

In this Section, we provide a succinct overview of the Chapters this Thesis is split into and also give some more information with respect to the Objectives described in Section 1.2. In particular:

- In Chapter 2, we provide to the reader all the necessary background knowledge that is of paramount importance to understand the Chapters that follow. More specifically, we begin with a brief introduction to CA techniques with a focus on deterministic (such as Principal Component Analysis (PCA) [31], Linear Discriminant Analysis (LDA) [32] and robust PCA [33]) as well as probabilistic approaches (such as Probabilistic LDA (PLDA) [8, 9]). Moreover, we introduce the concept of GANs [20] and detail the GAN methodologies that are closely related to our work, such as the conditional GAN variations [34, 35, 25] and various GAN loss/architectures such as Wasserstein GAN with

Gradient Penalty (WGAN-GP) [36, 37].

- In Chapter 3, we present our work "AgeDB: the first manually collected, in-the-wild age database", which constitutes the first introduced dataset in the literature that is annotated for multiple attributes such as identity, gender, and accurate to the year age of each person depicted in the image. The fact that AgeDB contains accurate to the year age labels which are manually annotated, makes it a very valuable benchmark for age estimation applications. Moreover, apart from the fact that AgeDB is manually collected, it contains images of the same celebrities depicted at different ages and it can thus be used in an extended range of age related experiments, such as age progression "in-the-wild" or age-invariant face recognition. As we also show, the fact that we have annotations for multiple attributes proves beneficial in the task of age progression: by exploiting all of the available attribute information, we can extract more meaningful subspaces to describe the different age groups. As we demonstrate in an extensive number of quantitative as well as qualitative experiments, AgeDB can indeed become the leading benchmark dataset for face analysis applications.

- In Chapter 4, we present our work "Multi-Attribute Robust Component Analysis for Facial UV Maps" [10], dubbed MARCA, which is a deterministic CA method, able to work in multi-attribute 2D facial data. As already mentioned, CA techniques are very efficient for very well aligned data. Therefore, in MARCA, we take advantage of the latest 3D face alignment methods to project the 2D facial images on 2D facial UV maps. Such UV data, as we showcase later in the Chapter, prove to be very suitable for CA methods. In MARCA, we introduce the first robust CA method in the literature that is able to exploit all of the label information during training, even in the scenario where the dataset is annotated for multiple attributes. To achieve that, we adopt ideas of earlier works in the literature such as the ones stemming from robust PCA [33] and provide all the closed form solutions and mathematical proofs. Finally, we demonstrate the

efficacy of MARCA against widely used CA counterparts in a wide range of experiments, including, but not limited to, synthetic experiments and image denoising when data are contaminated with different types of noise as well as facial image translation.

- In Chapter 5, we present our work "Multi-Attribute Probabilistic Linear Discriminant Analysis for 3D Facial Shapes" [11], dubbed MAPLDA, which is a probabilistic CA method, able to work with multi-attribute 3D facial data. 3D facial data/meshes can easily be registered using algorithms such as non-rigid ICP [38] under a common reference template. After the registration process is complete, all the 3D facial meshes are well aligned with each other and can thus be utilised by CA methods. In MAPLDA, we detail the first probabilistic CA framework that is able to operate in data annotated with respect to multiple attributes. In particular, MAPLDA exploits all attribute information during training, can conduct inferences about any subset of available attributes during testing, and can be efficiently used in a large battery of classification as well as generation tasks. More specifically, we showcase how MAPLDA can be used to enhance 3D face recognition with respect to the identity, age-group, weight-group or ethnicity attributes as well as demonstrate how we can use MAPLDA to generate new 3D facial data conditioned on any subset of the aforementioned categories.

- In Chapter 6, we present our work "3DFaceGAN: Adversarial Networks for 3D Face Representation and Translation", which is a deep learning framework especially tailored to 3D facial data/meshes. As briefly discussed before, 3D mesh data in general are notoriously difficult to be used by DL techniques because they are non-Euclidean and thus the 2D convolutional operator cannot be directly used on them. Recently, due to the advances in Geometric Deep Learning [27], some deep learning works have appeared in the literature that use mesh convolutions in order to be used in 3D data [29]. Nonetheless, the results in such works lack in detail. To alleviate that, instead of working directly on the registered meshes, in 3DFaceGAN, we first transform the 3D facial meshes into

2D facial UV maps. Having these 2D representations of the 3D facial data, we can use all the prevalent deep learning techniques utilised in 2D image generative tasks. More specifically, in 3DFaceGAN, we introduce a GAN methodology with a novel adversarial loss designed for 3D facial data generation/reconstruction. As we demonstrate in a series of experiments, 3DFaceGAN improves upon the state-of-the-art by a significant margin.

- Finally, in Chapter 7, we summarise the contents and draw conclusions. Moreover, we report limitations of the discussed works, propose possible extensions, as well as succinctly report future research avenues which can be based on the methods we detailed in the previous Chapters.

1.4 Impact and Applications

CA methods, although they can only capture linear correlations between data, prove to be very efficient in small sample size (SSS) problems. On the other hand, DL approaches can capture higher order correlations between data, but in order to be efficient, they need large quantities of data. Focusing on the advantages of both worlds, our methods can be used in large variety of both generative and recognition tasks, such as:

- **Biometrics**. All of the CA and DL methods we propose can be used to enhance the accuracy and performance of biometrics applications. In particular, as we also discuss later, our methods can be used for example to improve the Face ID applications on mobile phones and thus make them more secure or utilised in CCTV applications to enhance the quality of the captured images.

- **Computer Graphics**. The proposed DL methods can be used to improve the 3D face models of AR/VR applications or computer games, generate more detailed emojis on mobile devices, etc.

- **Health and Safety**. Modern applications in cars monitor the physical condition of the drivers and can detect signs of tiredness. The proposed CA algorithms can be used to detect such changes in the face and can thus be integrated in such frameworks.

- **Finance**. The prices of financial commodities such as stocks are dependent on multiple factors/attributes. PCA has long been used in finance as a tool to clean noisy data. The introduction of the deterministic or probabilistic formulations under the multi-attribute scenario, could also be deemed beneficial when cleaning such noisy, multi-attribute data.

- **Robotics**. As robots become increasingly involved in our lives and day-to-day activities, the proposed methods can be used to recognise people and/or facial expressions and, based on this, take decisions tailored to each individual.

1.5 Publications

In this Section, I provide the list of the publications I either authored or co-authored during the years I have been working as a PhD student at Imperial College London. In particular, I split the publications in two parts: a) the publications which are directly relevant to the Chapters of this Thesis, and b) the publications that I was involved in and are not covered in detail in this Thesis.

1.5.1 Relevant Publications

- **Stylianos Moschoglou**, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, Stefanos Zafeiriou. AgeDB: the first manually collected, in-the-wild age database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPR-W), 2017.*

- **Stylianos Moschoglou**, Stylianos Ploumpis, Mihalis Nicolaou, Stefanos Zafeiriou. Multi-Attribute Probabilistic Linear Discriminant Analysis for 3D Facial Shapes. *Proceedings*

of the Asian Conference on Computer Vision (ACCV), 2018.

- **Stylianos Moschoglou**, Evangelos Ververas, Yannis Panagakis, Mihalis Nicolaou, Stefanos Zafeiriou. Multi-Attribute Robust Component Analysis for Facial UV Maps. *IEEE Journal of Selected Topics in Signal Processing (STSP), 2018.*

- **Stylianos Moschoglou**, Stylianos Ploumpis, Mihalis Nicolaou, Athanasios Papaioannou, Stefanos Zafeiriou. 3DFaceGAN: Adversarial Nets for 3D Face Representation, Generation, and Translation. *International Journal of Computer Vision (IJCV), 2020.*

1.5.2 Other publications

- **Stylianos Moschoglou**, Mihalis Nicolaou, Yannis Panagakis, Stefanos Zafeiriou. Initializing Probabilistic Linear Discriminant Analysis. *Proceedings of the European Signal Processing Conference (EUSIPCO), 2017.*

- Stylianos Ploumpis, **Stylianos Moschoglou**, Vasilios Triantafyllou, Stefanos Zafeiriou. 3D human tongue reconstruction from single "in-the-wild" images. *to be submitted.*

- Grigorios Chrysos, **Stylianos Moschoglou**, Giorgos Bouritsas, Yannis Panagakis, Jiankang Deng, Stefanos Zafeiriou. P-Nets: Polynomial Neural Networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.*

- Alexandros Lattas, **Stylianos Moschoglou**, Baris Gecer, Stylianos Ploumpis, Vasilios Triantafyllou, Abijeet Ghosh, Stefanos Zafeiriou. AvatarMe: Realistically Renderable 3D Facial Reconstruction. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2020.*

- Grigorios Chrysos, **Stylianos Moschoglou**, Giorgos Bouritsas, Yannis Panagakis, Jiankang Deng, Stefanos Zafeiriou. Deep Polynomial Neural Networks. *Published on Arxiv, submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI), 2020.*

- Grigorios Chrysos, **Stylianos Moschoglou**, Yannis Panagakis, Stefanos Zafeiriou. PolyGAN: High-order Polynomial Generators. *Published on Arxiv, 2019.*

- Stylianos Ploumpis, Evangelos Ververas, Eimear O' Sullivan, **Stylianos Moschoglou**, Haoyang Wang, Nick Pears, William Smith, Baris Gecer, Stefanos Zafeiriou. Towards a complete 3D morphable model of the human head. *IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI), 2020.*

- Baris Gecer, Alexandros Lattas, Stylianos Ploumpis, Jiankang Deng, Athanasios Papaioannou, **Stylianos Moschoglou**, Stefanos Zafeiriou. Synthesizing coupled 3d face modalities by trunk-branch generative adversarial networks. *Proceedings of the European Conference on Computer Vision (ECCV), 2020.*

Chapter 2

Related Work

Contents

In the first Section of this Chapter, we present the UV unwrapping process of textures and shapes of 3D data, a projection mechanism which is important for Chapters 4 and 6. In the next four Sections we cover all the necessary material around CA algorithms which is essential in order to introduce later our contributions to the literature. From Section five onwards, we detail all the DL algorithms which lay the ground for our own work in the relevant literature and which we introduce in the following Chapters.

2.1 UV map unwrapping of textures and shapes of 3D meshes

UV maps are the most common way to store the texture information that belongs to a 3D mesh. The letter "U" stands for the horizontal axis while the letter "V" for the vertical one. More precisely, let us consider a 3D mesh with n vertices, $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$, such that $\boldsymbol{x}_i \in \mathbb{R}^3, \forall i \in \{1, \ldots, n\}$. The topology of the 3D mesh is represented by an ordered triangle list, $\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_m$, such that every triangle is defined by three distinct mesh vertices. These triangles make up the so called mesh "faces". The texture information is stored in a UV map, where every vertex $\boldsymbol{x}_i, i \in \{1, \ldots, n\}$, of the 3D mesh is mapped to a specific location $\boldsymbol{c}_i = \left[c_u^i, c_v^i\right] \in \mathbb{R}^2$ on the UV map. For the texture values of each 3D mesh triangle that belong to inter-vertex locations, a linear interpolation of the texture coordinates across the triangle is applied [39]. A graphical representation of this procedure is shown in Fig. 2.1.

Apart from storing the texture information of a 3D mesh, UV maps can also store the actual 3D shape information. In order to achieve this, instead of storing the RGB texture information in each UV coordinate, we store the XYZ coordinates of the vertices. By projecting the 3D shape information of the 3D mesh on a UV map, we can use operators such as the 2D convolution. In this way, as we show later in Chapter 6, we can achieve state-of-the-art results in 3D mesh representation tasks.

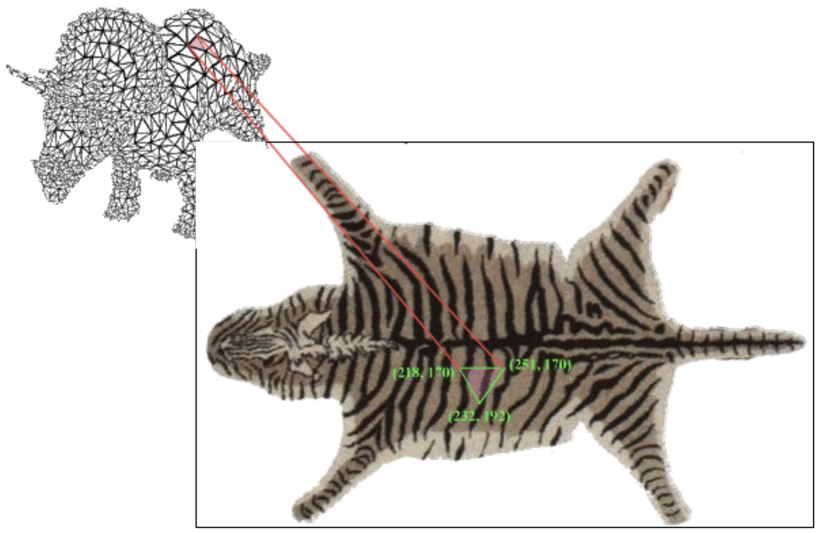


Figure 2.1: Texture mapping using UV maps. For each triangle in the 3D model, a corresponding set of UV coordinates is allocated. The figure has been adopted from the lectures of Dr Berhard Kainz' course "Computer Graphics" at Imperial College London. Dr Kainz has kindly provided his permission to reproduce it here.

2.2 Principal Component Analysis

Undoubtedly, the most widely used technique for dimensionality reduction in all sorts of data is the so-called Principal Component Analysis (PCA) [40, 41, 42]. By applying PCA on our data, we can extract from them the most essential information and thus by doing so we will hopefully eliminate all the unnecessary bits of information which are not useful for the problem at-hand. To achieve this, we try to find a subspace of lower dimensionality and on which when we project our data, most of the information or otherwise the maximum variance between them will be retained.

To structure the problem in the mathematical parlance, suppose we have a collection of N samples, i.e., $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, and that each of which lies in $\mathbb{R}^F$. PCA tries to find an orthogonal subspace of rank D, where $D < F$, and where the variance between the data is maximised. Suppose this orthogonal subspace is comprised of a set of projections that is

denoted as $\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_D \end{bmatrix} \in \mathbb{R}^{F \times D}$. Then, each of the N samples can be projected as $\mathbf{y}_i = \mathbf{W}^T \mathbf{x}_i, \forall i \in \{1, \dots, N\}$, such that $\mathbf{y}_i \in \mathbb{R}^D$.

In order to find the optimal set of orthogonal projections $\mathbf{W}$, we need to solve:

$$\mathbf{W}_o = \arg\max_{\mathbf{W}} \operatorname{tr}\left[\mathbf{W}^{\mathbf{T}}\mathbf{S}\mathbf{W}\right], \tag{2.1}$$
$$\text{subject to } \mathbf{W}^T\mathbf{W} = \mathbf{I},$$

where $\mathbf{S}$ is the covariance matrix defined by the samples provided. The constraint in (2.1) was put as otherwise the trivial solution to the problem would be ∞. By formulating the Lagrangian and solving it, we get:

$$\mathbf{S}\mathbf{W} = \mathbf{W}\mathbf{\Lambda} \tag{2.2}$$

Looking at (2.2), we can see that the optimal set of projections $\mathbf{W} \in \mathbb{R}^{F \times D}$ is given by the eigenvectors of $\mathbf{S}$ that correspond to the largest D non-zero eigenvalues. A visual illustration of PCA in 3D data is shown in Fig. 2.2, while the exact PCA steps are reported in Algorithm 1.

2.3 Robust Principal Component Analysis

Although PCA [40] which we described in the previous Section is very good and popular as a dimensionality reduction technique, it is not very robust when the data we have at our disposal have outliers and sparsities (some samples of such data are presented in Fig. 2.3). To address this, robust PCA [33, 43] was introduced in the literature, which is able to successfully handle such data.

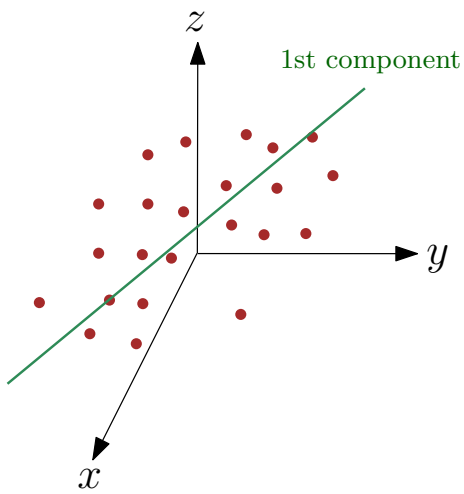


Figure 2.2: Visualisation of the first Principal Component when applying PCA on this 3D dataset. The Principal Component points in such direction that when the 3D data are projected on it, the variance among the data is maximised.

Algorithm 1 PCA algorithm

Input: A set of N data samples $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_N\end{bmatrix}$, with $\mathbf{x}_i \in \mathbb{R}^F, \forall i \in \{1, \dots, N\}$.
Output: A matrix $\mathbf{W} \in \mathbb{R}^{F \times D}$ which contains the first D Principal Components.

1. Center the data row-rise and denote the new matrix as $\overline{\mathbf{X}}$.
2. Calculate the covariance matrix $\mathbf{S} = \frac{1}{N}\overline{\mathbf{X}}\overline{\mathbf{X}}^T$.
3. Perform eigen-analysis on $\mathbf{S}$ and sort the eigenvectors in a decreasing fashion by looking at the corresponding eigenvalues.
4. Pick the first D eigenvectors and define a new matrix $\mathbf{W} = \begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_D\end{bmatrix}$.
5. Bonus step: to perform PCA on a random sample $\mathbf{x}_t$, we need to calculate $\mathbf{y}_t = \mathbf{W}^T\mathbf{x}_t$, where $\mathbf{y}_t$ is the projection we wanted to find.

Suppose that we have a data matrix $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_N\end{bmatrix}$, which we already know it is comprised of data that are sparse, perturbed with non-Gaussian noise, etc. Our aim is to extract a low-rank component $\mathbf{L}_0$, which contains the most useful information. As a result, let us assume the following decomposition:

$$\mathbf{X} = \mathbf{L}_o + \mathbf{S}_o, \tag{2.3}$$

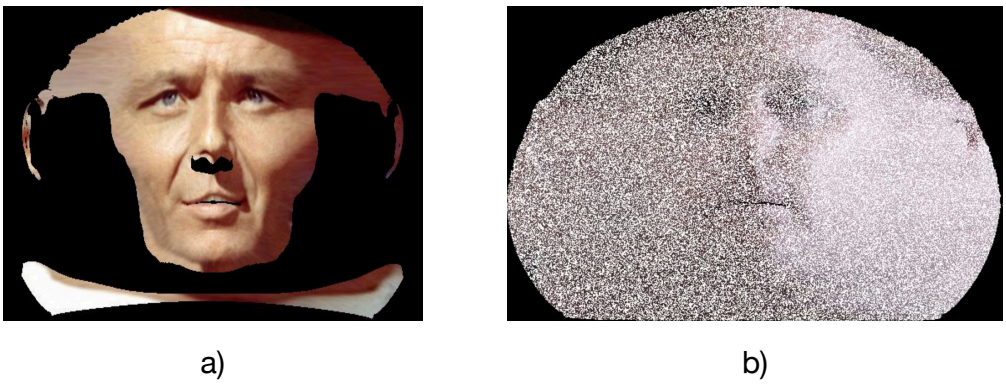


Figure 2.3: When working with data that have missing values, such as the one presented in a) or noisy data in which the noise is non-Gaussian, such as the one presented in b), PCA [40] would fail to work properly as it is sensitive to outliers. Under these scenarios, i.e., when we are dealing with sparsities in the data, approaches such as Robust PCA [33, 43] should be considered. The samples have been taken from our work MARCA [10].

where $\mathbf{L}_0$ is the row-rank component and $\mathbf{S}_0$ is the sparse one. Moreover, we do not possess a priori any knowledge about the row-space of $\mathbf{L}_0$ or the positions of the non-zero elements in $\mathbf{S}_0$ or even how many they are. The task of robust PCA [33] is to extract as accurately and efficiently as possible this low-rank component. To achieve this, let us consider the following problem:

$$\begin{aligned} &\text{minimise } \|\mathbf{L}\| + \gamma \|\mathbf{S}\|, \\ &\text{subject to } \mathbf{L} + \mathbf{S} = \mathbf{X}. \end{aligned} \tag{2.4}$$

To solve (2.4), we employ the Alternating Direction Method of Multipliers (ADMM) [44, 45] algorithm. After formulating the Augmented Lagrangian and solving individually for the updates of the parameters, we get the final solution according to Algorithm 2.

In Algorithm 2, SVT (Singular Value Thresholding) is defined as follows:

$$\mathrm{SVT}_\lambda(\mathbf{M}) = \mathbf{U}\mathcal{D}_\lambda(\mathbf{\Sigma})\mathbf{V}^T, \quad \mathcal{D}_\lambda(\sigma) = \mathrm{diag}\left(\mathrm{sgn}(\sigma) \cdot \max(\|\sigma\| - \lambda)\right), \tag{2.5}$$

Algorithm 2 ADMM solver for robust PCA

Input: The data matrix $\mathbf{X}$ which is comprised of the input data.
Initialisations: Random initialisations for $\mathbf{S}$, the Lagrangian $\mathbf{\Lambda}$. Set the initial time step $t = 1$ and the penalty parameter at a random positive value $\lambda > 0$.
Output: The optimal low-rank component $\mathbf{L}_o$ and optimal sparse component $\mathbf{S}_o$.
while *convergence criterion not met* **do**

1. $\mathbf{L}_t \leftarrow \mathrm{SVT}_\lambda \left(\mathbf{X} - \mathbf{S}_{t-1} + \mathbf{\Lambda}_{t-1}\right)$
2. $\mathbf{S}_t \leftarrow \mathrm{ET}_{\gamma\lambda} \left(\mathbf{X} - \mathbf{L}_t + \mathbf{\Lambda}_{t-1}\right)$
3. $\mathbf{\Lambda}_t \leftarrow \mathbf{\Lambda}_{t-1} + \frac{1}{\lambda} \left(\mathbf{X} - \mathbf{L}_t - \mathbf{S}_t\right)$
4. $t \leftarrow t + 1$

end

where $\mathbf{M} = \mathbf{U\Sigma V}^T$ is the Singular Value Decomposition (SVD) of matrix $\mathbf{M}$. Also, ET (Entry-wise soft Thresholding) is defined as follows:

$$\left(\mathrm{ET}_{\gamma\lambda}\left(\mathbf{M}\right)\right)_{ij} = \begin{cases} M_{ij} - \gamma\lambda & \text{if } M_{ij} \geq \lambda\gamma, \\ 0 & \text{if } -\lambda\gamma \leq \|M_{ij}\| \leq \lambda\gamma, \\ M_{ij} + \lambda\gamma & \text{if } M_{ij} \leq -\lambda\gamma. \end{cases} \tag{2.6}$$

Finally, in Fig. 2.4, we demonstrate a typical application of robust PCA [33].

2.4 Linear Discriminant Analysis

The CA algorithms that we have presented thus far like PCA [40] and robust PCA [33] are all *unsupervised*, meaning that they do not take into consideration any label information that might exist during the training process. Nevertheless, there is a significant number of applications where it is important to take into account the label information, as we might for example want to extract information only about a certain class of data and not statistics about the dataset in its entirety. Moreover, as we will also show later, by taking into consideration

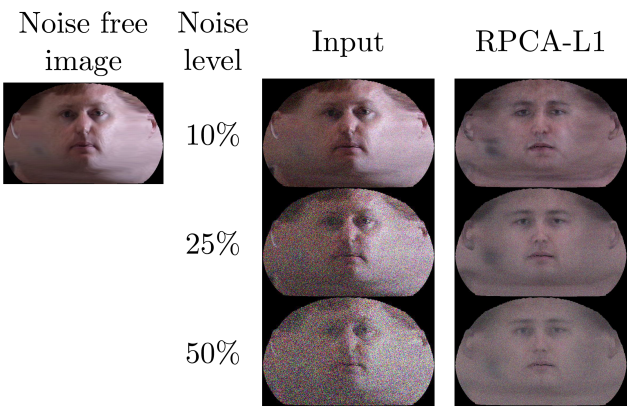


Figure 2.4: Example of a typical robust PCA [33] denoising application, utilising a variant of robust PCA [46]. As can be seen, robust PCA [46] is able to successfully extract the denoised component of interest (the low-rank) and eliminate the redundant information (i.e., the noise). The figure is taken from our work MARCA [10].

the information provided by the labels, we can increase the performance of the algorithms in the tasks such as 2D image translation and 3D face identification.

Adamantly, the most widely used *supervised* CA algorithm is Linear Discriminant Analysis (LDA) [32, 47]. In particular, when we apply LDA on our labelled dataset, we try to make the data that belong in each class to look as similar as possible, whereas the data that belong in different classes to look as dissimilar as possible. To achieve that, we: a) try to minimise the variability (i.e., the variance) in each class, and b) maximise the distance between the data that belong in different classes (by increasing the distance of the classes' means).

In the mathematical parlance, suppose we have a collection of N samples, i.e., $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$, and that each of which lies in $\mathbb{R}^F$. Suppose also that our data are annotated and split into a total of C classes. In order to solve the optimisation problem, we need to define two matrices, the so-called within-class scatter matrix, which measures the variability within each class, and the so-called between-class scatter matrix, which measures the distance between the classes. As a result, we have:

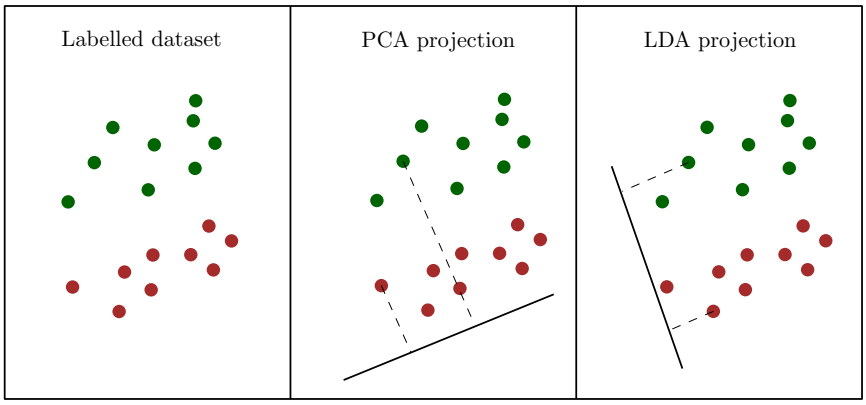


Figure 2.5: LDA [32] vs PCA [40]. As evinced in the graph, PCA does not take into account the label information and merely tries to maximise the variance of the data on the projected subspace. On the other hand, LDA does take into consideration the labels and thus: a) maximises the distance among the data that belong in different classes, b) brings the data that belong in the same class closer to each other.

$$\mathbf{S}_w = \sum_{j=1}^{C} \frac{1}{N_{c_j}} \sum_{\mathbf{x}_i \in c_j} \left(\mathbf{x}_i - \boldsymbol{\mu}_{c_j}\right) \left(\mathbf{x}_i - \boldsymbol{\mu}_{c_j}\right)^T, \tag{2.7}$$

$$\mathbf{S}_b = \sum_{j=1}^{C} \left(\boldsymbol{\mu}_{c_j} - \boldsymbol{\mu}\right) \left(\boldsymbol{\mu}_{c_j} - \boldsymbol{\mu}\right)^T, \tag{2.8}$$

where $N_{c_j}, \forall j \in \{1, \ldots, C\}$, denotes the number of data in each class j, $\boldsymbol{\mu}_{c_j}$ the mean of the data that belong in class j, and $\boldsymbol{\mu}$ the global mean of the dataset in its entirety.

Having defined the helper matrices, the optimisation problem can be constructed as follows:

$$\mathbf{W}_o = \arg\max_{\mathbf{W}} \operatorname{tr}\left[\mathbf{W}^{\mathbf{T}} \mathbf{S}_b \mathbf{W}\right], \tag{2.9}$$

$$\text{subject to } \mathbf{W}^T \mathbf{S}_w \mathbf{W} = \mathbf{I}.$$

In other words, 2.9 tries to maximise the distance between the classes subject to the constraint that the variability between the data is not distorted. By formulating the Lagrangian and solving it, we get:

$$\mathbf{S_b W} = \mathbf{S}_w \mathbf{W} \mathbf{\Lambda} \tag{2.10}$$

Looking at (2.10), we can see that the optimal set of projections $\mathbf{W} \in \mathbb{R}^{F \times D}, D < C$ is given by the eigenvectors of $\mathbf{S}_w^{-1}\mathbf{S}_b$ that correspond to the largest D non-zero eigenvalues. A visual illustration of LDA in 3D data is shown in Fig. 2.5, while the exact LDA steps are reported in Algorithm 3.

Algorithm 3 LDA algorithm

Input: A set of N data samples $\mathbf{X} = \begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_N\end{bmatrix}$, with $\mathbf{x}_i \in \mathbb{R}^F, \forall i \in \{1, \dots, N\}$, where each datum is annotated with a class label $c, c \in \{1, \dots, C\}$.
Output: A matrix $\mathbf{W} \in \mathbb{R}^{F \times D}$ which contains the first D LDA components.

1. Compute the within-class scatter matrix $\mathbf{S}_w$ as defined in (2.7).
2. Compute the between-class scatter matrix $\mathbf{S}_b$ as defined in (2.8).
3. Perform eigen-analysis on $\mathbf{S}_w^{-1}\mathbf{S}_b$ and sort the eigenvectors in a decreasing fashion by looking at the corresponding eigenvalues.
4. Pick the first D eigenvectors and define a new matrix $\mathbf{W} = \begin{bmatrix}\mathbf{w}_1 & \mathbf{w}_2 & \dots & \mathbf{w}_D\end{bmatrix}$.

2.5 Probabilistic Linear Discriminant Analysis

Although LDA [32] is a broadly used technique in discriminative tasks when we have labelled data at our disposal, it does not behave well in certain tasks such as face recognition of subjects whose pictures may vary in pose or lighting [9]. Moreover, LDA is a deterministic CA algorithm and, compared to probabilistic CA approaches, it is not as flexible to take into consideration uncertainty factors.

To alleviate all the aforementioned limitations, Prince et al. introduced Probablistic LDA (PLDA) [9, 48] in the literature. PLDA can be considered as a probabilistic variant of LDA, as it is a CA algorithm that can work with labelled data and has the probabilistic element at its core.

To put the words into a mathematical context, PLDA carries the assumption that data are generated by two different subspaces: one that depends on the class and one that depends on the sample. That is, assuming we have a total of I classes, with each class i containing a total of J samples, then the j-th datum of the i-th class is defined as:

$$\mathbf{x}_{ij} = \boldsymbol{\mu} + \mathbf{F}\mathbf{h}_i + \mathbf{G}\mathbf{w}_{ij} + \boldsymbol{\epsilon}_{ij}, \tag{2.11}$$

where $\boldsymbol{\mu}$ denotes the global mean of the training set and $\mathbf{F}$ defines the subspace capturing the classes of the data, with $\mathbf{h}_i$ being the latent variable representing the specific position in the particular subspace. As a result, term $\mathbf{F}\mathbf{h}_i$ in its entirely provides the specific class in which datum $\mathbf{x}_{ij}$ belongs. Furthermore, $\mathbf{G}$ defines the subspace modelling variations among data of the same class, with $\mathbf{w}_{ij}$ being the associated latent variable for the specific datum. As a result, term $\mathbf{G}\mathbf{w}_{ij}$ in its entirely provides the specific variations that characterise the said datum. Finally, $\boldsymbol{\epsilon}_{ij}$ denotes a residual noise term which is Gaussian with diagonal covariance $\boldsymbol{\Sigma}$ and is used to explain the rest of the information which can not be captured by the other two components.

Assuming zero-mean observations, the model in (2.11) can be described as:

$$P\left(\mathbf{x}_{ij}|\mathbf{h}_i, \boldsymbol{w}_{ij}, \boldsymbol{\theta}\right) = \mathcal{N}_{\mathbf{x}}\left(\mathbf{F}\mathbf{h}_i + \mathbf{G}\mathbf{w}_{ij}, \boldsymbol{\Sigma}\right), \tag{2.12}$$

$$P\left(\mathbf{h}_i\right) = \mathcal{N}_{\mathbf{h}}\left(\mathbf{0}, \mathbf{I}\right), \tag{2.13}$$

$$P\left(\mathbf{w}_{ij}\right) = \mathcal{N}_{\mathbf{w}}\left(\mathbf{0}, \mathbf{I}\right), \tag{2.14}$$

where the set of parameters $\boldsymbol{\theta} = \{\mathbf{F}, \mathbf{G}, \boldsymbol{\Sigma}\}$ is optimised during training via the EM algorithm [49, 50]. After the training process is complete (e.g., when a number of iterations has finished or when a certain threshold in the error of the validation set has been met), the optimal set of parameters $\boldsymbol{\theta} = \{\mathbf{F}, \mathbf{G}, \boldsymbol{\Sigma}\}$ is found. During the testing process, the optimal subspaces derived in the training process are utilised in order to extract the most probable class in which the datum belongs. In Fig. 2.6, we demonstrate an application of PLDA in the task of 3D face

ethnicity recognition, where visualisations of the components of the various ethnicities are provided.

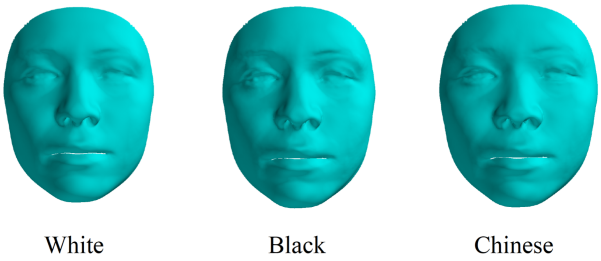


Figure 2.6: Ethnicity components of PLDA model in a 3D face ethnicity recognition application. In this task, we utilise PLDA to learn the subspace that pertains to the ethnicity attribute (i.e., White, Black, and Chinese in our case) and the corresponding ethnicity latent variables that produce the ethnicity components depicted above. By using PLDA, we are able to learn the subspace of interest (i.e., the ethnicity one) and use only this during testing (thus effectively avoiding to take into account information that is not needed during inference, such as the individual characteristics of the person depicted in 3D). The figure has been taken from our work MAPLDA [11].

2.6 Autoencoders

As we have previously pointed out, PCA [40] is a popular dimensionality reduction technique but one of its main drawbacks is that it can only capture linear relationships between data, failing to take into consideration higher order relationships. However, with the introduction of DL and the increasing availability of computational resources, it was possible to create DL architectures that can capture higher order relationships between data.

One of the most popular DL approaches that are used for dimensionality reduction as well as data reconstruction is a type of Artificial Neural Networks (ANN) called AutoEncoders (AEs) [51, 52]. In particular, an AE typically consists of three parts, namely:

- the *encoder*, which is the one that receives the input datum. The encoder is comprised of a number of layers whose main purpose is to gradually reduce the dimensionality of the input datum.

- the *latent embedding*, which is the encoded version of the input datum. The latent embedding is typically a 1D vector and it is the output of the encoder. The dimensionality of the latent embedding is significantly reduced compared to the one of the input datum.
- the *decoder*, which receives as input the latent embedding and is comprised of a number of layers which gradually increase the dimensionality of its input till it is exactly the same as the one of the original datum.

Depending on the task the AE is asked to carry out, various names and implementations exist in the literature, such as denoising AE [52], variational AE [53], sparse AE [54], contractive AE [55], convolutional mesh AE [29], etc. A graphical illustration of the different parts of an AE is shown in Fig. 2.7.

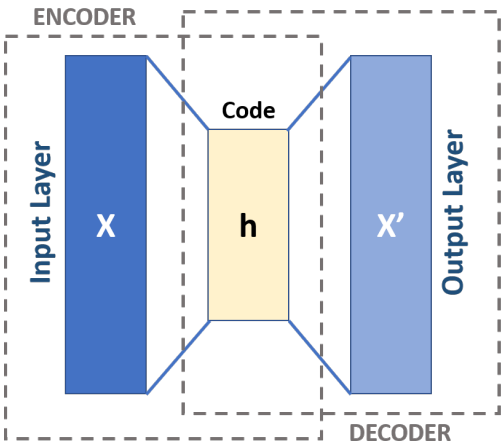


Figure 2.7: Typical structure of an AE [51]. The encoder receives the input datum (denoted as $\mathbf{x}$) and encodes it into a latent embedding denoted as $\mathbf{h}$ (referred to also as "code" in the figure). The decoder receives as input the latent embedding $\mathbf{h}$ and tries to reconstruct the original datum as accurately as possible (the output of the decoder is denoted as $\mathbf{x}'$). The figure has been taken from Wikipedia and was created by Michela Massi. It is provided under CC BY-SA 4.0 license and thus can be replicated here (https://bit.ly/31Mfe2T).

In the mathematical parlance, without loss of generality, let us consider an AE that is comprised of one layer in the encoder and another one in the decoder parts, respectively. If we denote with $\mathbf{x}$ the input datum, $\mathbf{W}$ the weights matrix and $\mathbf{b}$ the biases of the encoder, $\mathbf{W}'$

the weights matrix and $\mathbf{b}'$ the biases of the decoder, σ the activation function, $\mathbf{h}$ the latent embedding, and $\mathbf{x}'$ the output of the decoder (or the reconstructed datum), the whole process can be described as:

$$\mathbf{h} = \sigma\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right), \tag{2.15}$$

$$\mathbf{x}' = \sigma\left(\mathbf{W}'\mathbf{h} + \mathbf{b}\right)'. \tag{2.16}$$

Combining (2.15) and (2.16), we get:

$$\mathbf{x}' = \sigma\left(\mathbf{W}'\left(\sigma\left(\mathbf{W}\mathbf{x} + \mathbf{b}\right)\right) + \mathbf{b}'\right). \tag{2.17}$$

Since the goal of an AE is to successfully reconstruct the input data, the standard loss to be optimised during the training is the least-squares one, which is defined as follows:

$$\mathcal{L}\left(\mathbf{x}, \mathbf{x}'\right) = \left\|\mathbf{x} - \mathbf{x}'\right\|^2, \tag{2.18}$$

where the total set of parameters $\boldsymbol{\theta} = \{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{b}'\}$ (implicitly included in the term $\mathbf{x}'$ as shown in (2.17)) is optimised during training via back-propagation [21], as normally happens in a typical ANN. Having found the optimal set of parameters, the latent embedding is given by (2.15), whereas the reconstruction is provided by (2.17). In Fig. 2.8, we show some reconstructions of MNIST [56] digits when we utilise a standard convolutional AE. We should point out that depending on the application, the loss function that is to be optimised can change and include other terms as well such as regularisers [54], etc.

Figure 2.8: Reconstruction of MNIST digits [56], utilising a standard convolutional AE trained with TensorFlow [57]. As evinced, the AE is able to capture non-linear relationships between the data and thus the reconstructions look quite sharp, with an increasing level of detail. However, they are still not completely realistic (we refer the reader to the next Section which explains how we can further improve the generated image quality). The figure has been taken from Keras blog and was created by Francois Chollet. Keras is under MIT license and thus the figure can be reproduced here (https://bit.ly/3lOWPKO).

2.6.1 Convolutional Mesh AE

AEs have been broadly used in 2D image data because, as we have mentioned, operators such as the convolution work well in Euclidean data. Nevertheless, since 3D data are not Euclidean, the current implementations of AEs cannot work "as-is" on 3D data.

To alleviate this, Ranjan et al. introduced the Convolutional Mesh Autoencoder [29], dubbed as CoMA, which is the first AE in the literature that is able to work on 3D data, exploiting recently introduced techniques in the field of Geometric Deep Learning [27], such as spectral convolutions [58].

More specifically and without delving too much into the mathematical details, the authors do mesh filtering using the idea of recursive Chebyshev polynomials [58] in the Fourier domain. Moreover, to keep the local as well as the global structure of the 3D data, the authors do a hierarchical down/up sampling of the resolution of the mesh utilising quadric matrices [59].

During training, the parameters of the Chebyshev polynomials are optimised via back-propagation [21]. The preferred loss function which is used to minimise the error between the input and reconstructed 3D data is the $L1$ loss one. Reconstructed samples utilising CoMA

Figure 2.9: Reconstructions of 3D data utilising CoMA. As can be seen, although CoMA can capture higher-order relationships between the data, it still lacks in capturing finer facial details. The figure have been taken from our work 3DFaceGAN [30].

are presented in Fig. 2.9.

2.7 Generative Adversarial Networks

Generative Adversarial Networks (GANs) [20] have undoubtedly revolutionised the whole field of Computer Vision, especially in the generative tasks, whose performance reached unprecedented levels over the last couple of years. The main advantage of GANs compared to their counterparts is that they can approximate the underlying distribution of the dataset upon which are trained in a very accurate way.

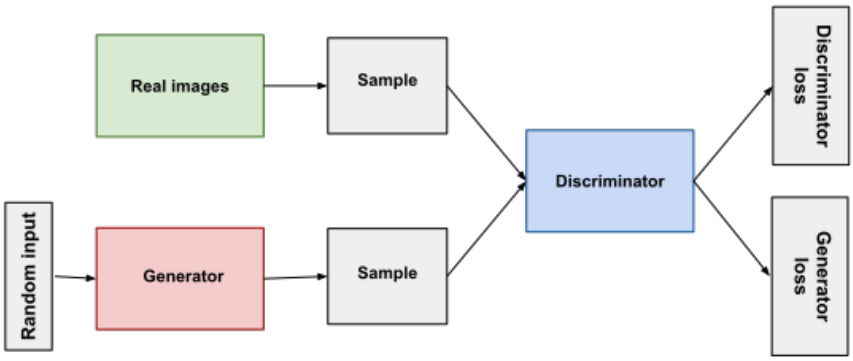


Figure 2.10: A standard GAN architecture that depicts the tasks of the generator as well as the discriminator. The figure have been taken from Google developers platform. It is under under CC BY-SA 4.0 license and thus it can be reproduced here (https://bit.ly/2EM5dKn).

GANs normally consist of two deep networks [20], namely the generator or G that generates the data and the discriminator or D that tries to understand whether the data that is fed with are either real (i.e., coming from the actual distribution) or generated (i.e., the ones that are produced from the generator). As the training procedure progresses, the samples that are produced from the generator are of increasing quality and it becomes harder and harder for the discriminator to discriminate the real from the fake (i.e., generated) data. By the end of the training session, the discriminator is ideally totally incapable of distinguishing the data and thus the generator is able to produce diverse data which look very realistic and with an increasing level of detail.

The huge success of GANs is attributed to the fact that the discriminator acts as a "teacher", so during back-propagation, the updates of the weights of the generator are adjusted according to the outputs of the discriminator. Moreover, since the discriminator itself is a deep network and determines the behaviour of the whole training process, we can avoid using "hand-crafted" loss functions and let the deep network do this for us. The whole procedure is called adversarial because the generator and the discriminator have competing goals: on the hand, the generator is trying to "fool" the discriminator by producing more realistic data and on the other, the discriminator tries to distinguish the real from the fake data.

Putting the words in a mathematical context, let us the denote the distribution of the real data as p_{data} and the distribution of the generated data as p_g. Furthermore, we define a prior $p_{\mathbf{z}}$ on the input noise $\mathbf{z}$ that is fed to the generator. We also denote the output of the generator as $G(\mathbf{z}; \boldsymbol{\theta}_g)$. In general, the generator G is a differentiable function that is comprised of a number of layers and activation functions, the parameters of which (containing all the weights and biases) are denoted as $\boldsymbol{\theta}_g$. Moreover, we denote the output of the discriminator as $D(\mathbf{x}; \boldsymbol{\theta}_d)$. The discriminator is also a differential function that is comprised of a number of layers and activation functions, the parameters of which (containg all the weights and biases) are denoted as $\boldsymbol{\theta}_d$. We should also note that $D(\mathbf{x}; \boldsymbol{\theta}_d)$ refers to the output of the discriminator

Algorithm 4 Training a GAN [20].

Input: The training data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$.
Initialisations: Random initialisations of the parameters of both G and$D$ following e.g., the method by He et al. [60]. k is a hyperparameter that refers to the number of steps we train D before continuing with G. m is a hyperparameter that refers to the total number of iterations during training. b is a hyperparameter that denotes the batch-size during training.
Output: The optimal D and G so that G is able to generate data that D can longer distinguish from the real ones.

for m steps **do**
  for k steps **do**
    - Sample b noise inputs $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(b)}\}$ from $p_g(\boldsymbol{z})$.
    - Sample b examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(b)}\}$ from $p_{\text{data}}(\boldsymbol{x})$.
    - Update the discriminator according to (2.19):

$$\nabla_{\theta_d} \frac{1}{b} \sum_{i=1}^{b} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

  end for
  - Sample b noise inputs $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(b)}\}$ from $p_g(\boldsymbol{z})$.
  - Update the generator according to (2.19):

$$\nabla_{\theta_g} \frac{1}{b} \sum_{i=1}^{b} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

when it is fed with a real datum $\mathbf{x}$.

As we described before, the discriminator D tries to maximise the probability that the correct label (real/fake) is assigned to the data that is fed with, whereas the generator G tries to make this impossible by constantly improving the quality of the generated/fake data. This can be mathematically formulated as:

$$\min_G \max_D V(G, D) = \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))]. \tag{2.19}$$

By looking at (2.19), we can see the competition between the generator and the discriminator as the one is trying to minimise the objective whereas the other one tries to maximise it. The

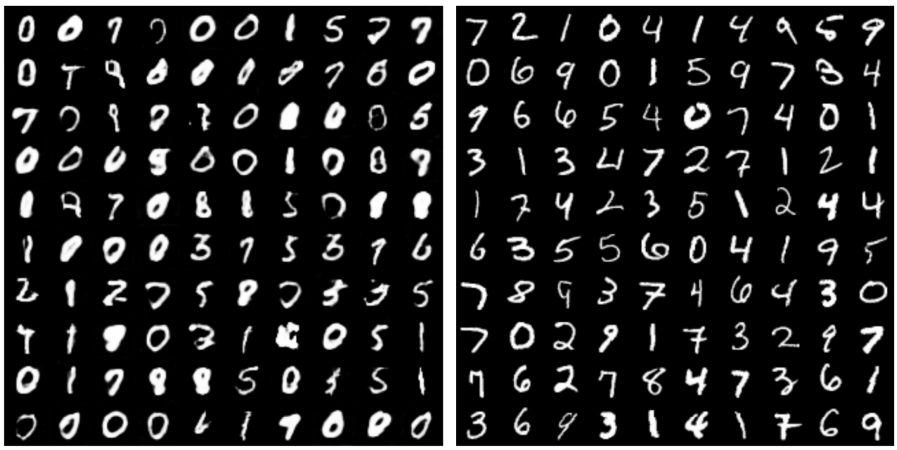


Figure 2.11: MNIST digits generated by a GAN [61]. As can be seen, the GAN is able to generate very realistic images which is hard to distinguish from the real data. The figure has been taken from our work PolyGAN [61].

algorithm for training the GAN is presented in Algorithm 4. A standard GAN architecture is shown in Fig. 2.10, whereas some samples generated by a GAN are depicted in Fig. 2.11.

Since the introduction of the first GAN [20], a myriad of different variations have been introduced in the literature. Some of those approaches have focused on the GAN architectures [62, 63, 22, 23, 24] and others on the loss function [36, 37, 64, 65]. In what follows, we will briefly report some of the GAN variations that are most relevant to our work.

2.7.1 Wasserstein GAN

Although the original GAN [20] was proven to be very good in generating realistic and sharp images, it did come with some flaws. One main problem is that quite frequently the generator's behaviour would lead to *mode collapse* (i.e., the generator being able to only generate a certain type (mode) of images and thus not providing diverse enough images). Another problem is that quite often the generator fails to learn any meaningful distributions and generates random noise throughout the whole training process. This set of drawbacks is primarily attributed to

the choice of the loss function, i.e., the Jensen-Shannon (JS) one [20], while fails to converge under certain scenarios [36].

To remedy the aforementioned shortcomings, Arjovsky et al. proposed the Wasserstein GAN [36], where the main contribution is that instead of utilising the Jensen-Shannon divergence as a loss function, they utilised the Earth Mover's (EM) distance. In layman's terms, supposing we have two distributions q and p, EM distance refers to the minimum or optimal "transport cost" so that q and p are alike. Using EM distance instead of the JS divergence stabilises the training and significantly reduces the occurrence of mode collapse during the training of a GAN, as EM distance is able to converge where JS fails to do so [36].

In mathematical terms, supposing that the distribution of the generator is denoted as $\mathbb{P}_g$ and the distribution of the real data as $\mathbb{P}_r$, the EM distance is defined as:

$$W\left(\mathbb{P}_r, \mathbb{P}_g\right) = \inf_{\gamma \in \Pi(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(\mathbf{x},\mathbf{y}) \sim \gamma}\left[\|\mathbf{x} - \mathbf{y}\|\right], \tag{2.20}$$

where $\Pi\left(\mathbb{P}_r, \mathbb{P}_g\right)$ denotes the set that contains all the joint distributions $\gamma\left(x, y\right)$, the marginals of which are $\mathbb{P}_r$ and $\mathbb{P}_g$, respectively.

The main issue with (2.20) is that the infimum is highly intractable. Nevertheless, the Kantorovich-Rubinstein duality [66], states that:

$$W\left(\mathbb{P}_r, \mathbb{P}_\theta\right) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_r}\left[f\left(\mathbf{x}\right)\right] - \mathbb{E}_{\mathbf{x} \sim \mathbb{P}_\theta}\left[f\left(\mathbf{x}\right)\right]. \tag{2.21}$$

We should note that the supremum in (2.21) is over every 1-Lipschitz function. If we had $\|f\|_L \leq K$ (K-Lipschitz continuity) in (2.21), we would have $K \cdot W\left(\mathbb{P}_r, \mathbb{P}_\theta\right)$. Therefore, for a parametrisable function $f_{\mathbf{w}}$ that is K-Lipschitz for some K, we could solve the following:

$$\max_{\mathbf{w}} \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_r}\left[f_{\mathbf{w}}\left(\mathbf{x}\right)\right] - \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[f_{\mathbf{w}}\left(g_{\boldsymbol{\theta}}\left(\mathbf{z}\right)\right)\right]. \tag{2.22}$$

In order to solve the maximisation problem of (2.21), we can a employ a neural network (GAN) with weights $\mathbf{w}$, as the authors showcase in [36]. In order to make sure that the functions $f_{\mathbf{w}}$ are K-Lipschitz for some K, we have to make sure the weights $\mathbf{w}$ are compact so we need to clip them after each gradient update. The Wasserstein GAN method is compactly presented in Algorithm 5.

Algorithm 5 Training WGAN [36]

Input: The training data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$.
Initialisations: λ, the learning rate. k, the clipping parameter. b, the batch size. m, the number of iterations of the critic per generator iteration. $\mathbf{w}_0$, initial critic parameters. $\boldsymbol{\theta}_0$, initial generator's parameters, t the total number of iterations.
Output: The optimal parameters $\mathbf{w}$ and $\boldsymbol{\theta}$.

```
for t steps do
  for m steps do
    Sample {x^(i)}_{i=1}^b ~ P_r a batch from the real data.
    Sample {z^(i)}_{i=1}^b ~ p(z) a batch of prior samples.
    g_w ← ∇_w [ (1/b) Σ_{i=1}^b f_w(x^(i)) − (1/b) Σ_{i=1}^b f_w(g_θ(z^(i))) ]
    w ← w + λ · backprop(w, g_w)
    w ← clip(w, −k, k) element-wise clipping
  end for
  Sample {z^(i)}_{i=1}^b ~ p(z) a batch of prior samples.
  g_θ ← −∇_θ (1/b) Σ_{i=1}^b f_w(g_θ(z^(i)))
  θ ← θ − λ · backprop(θ, g_θ)
end for
```

2.7.2 Wasserstein GAN with Gradient Penalty

One of the main problems of the Wasserstein GAN [36] is the clipping of the weights, which is used to make sure the function that is optimised is a K-Lipschitz one. This introduces several problems such as failure to converge in certain cases and a laborious effort to correctly identify the right clipping threshold [37].

In a sense, the weight clipping introduces some regularisation in the network so that the network weights adhere to a certain window. In [37], instead of using weight clipping, the authors introduce the concept of gradient penalty in the weights. In particular, it can be shown that a differentiable function f is 1-Lipschitz if and only if it has gradients with norm at most 1 everywhere [37]. As a result, in [37], the authors penalise the model if the norm of the gradients exceeds the value of 1:

$$\mathcal{L} = \underbrace{\mathbb{E}_{\tilde{\mathbf{x}}\sim\mathbb{P}_g}\left[D\left(\tilde{\mathbf{x}}\right)\right] - \mathbb{E}_{\mathbf{x}\sim\mathbb{P}_r}\left[D\left(\mathbf{x}\right)\right]}_{\text{original Wasserstein loss}} + \underbrace{\lambda \cdot \mathbb{E}_{\hat{\mathbf{x}}\sim\mathbb{P}_{\hat{\mathbf{x}}}}\left[\left(\left\|\nabla_{\hat{\mathbf{x}}} D\left(\hat{\mathbf{x}}\right)\right\|_2 - 1\right)^2\right]}_{\text{gradient penalty}}. \tag{2.23}$$

In (2.23), $\hat{\mathbf{x}}$ is sampled between $\mathbf{x}$ and $\tilde{\mathbf{x}}$ such that:

$$\hat{\mathbf{x}} = t\tilde{\mathbf{x}} + (1-t)\,\mathbf{x}, \tag{2.24}$$

where $t \in [0, 1]$ and is sampled uniformly.

2.7.3 Boundary Equilibrium GAN

Despite the improvements and stabilisations in GAN training that were introduced and we reported in the previous Sections, GANs were not able to produce realistic images when trained in more complex datasets that contain images of higher resolution. The first work in the literature that proved to be able to generate images of higher resolution with an increasing level of detail was the Boundary Equilibrium GAN (BEGAN) [63].

The main characteristic of BEGAN is that it introduces the equilibrium concept, where the behaviour of the discriminator is balanced against the behaviour of the generator (so that the training process is kept under control and the discriminator does not "win" relatively easily and thus hindering the performance of the generator). Moreover, based on the idea of EBGAN

[67], the discriminator in BEGAN has the form of an autoencoder. By utilising an autoencoder in the discriminator, we are able to match the (reconstruction) losses between the samples and not the samples themselves, which, as shown experimentally, improves the performance [63]. Overall, the objective that is optimised is as follows:

$$\mathcal{L}_D = \mathcal{L}(\mathbf{x}) - k_t \cdot \mathcal{L}(G(\mathbf{z})), \tag{2.25}$$

$$\mathcal{L}_G = \mathcal{L}(G(\mathbf{z})), \tag{2.26}$$

$$k_{t+1} = k_t + \lambda_k (\gamma \mathcal{L}(\mathbf{x}) - \mathcal{L}(G(\mathbf{z}))), \tag{2.27}$$

where (2.25) is the discriminator objective and (2.26) is the generator objective. The loss $\mathcal{L}$ refers to the $L1$ loss between the input and the output of the discriminator. As the authors show in [63], the losses introduced in (2.25) and (2.26) are similar to the Wasserstein GAN objective [36], the main difference being that we do not have to explicitly employ the Kantorovich-Rubinstein duality theorem [66]. Finally, the equilibrium constraint is introduced via the hyper-parameter $\gamma \in [0, 1]$ in (2.27), which controls how much emphasis should be put on the discriminator during training. In order to control the equilibrium, the authors in [63] use concepts of Proportional Control Theory. More specifically, they use a variable $k_t \in [0, 1]$ in (2.27), which keeps under control how much power is given on $\mathcal{L}(G(\mathbf{z}))$ during the gradient descent. λ_k is the "learning rate" for the variable k_t. The authors initiliase $k_0 = 0$ and as far as the learning rate λ_k is concerned, this depends on the dataset but the value is always relatively small (such as $10e - 3$).

2.7.4 Progressive GAN

Even though with the introduction of BEGAN [63] in the literature many problems revolving around GANs were alleviated, especially with regards to producing images of higher quality, BEGAN was still only able to generate images of good quality up to the size of 256×256.

Further increasing the resolution of the training images would lead to problems in the training such as frequent inability to converge, very slow behaviour in the training due to the extreme number of parameters, etc. To remedy such problems, Progressive GAN [23] was introduced in the literature.

Figure 2.12: Generations of facial images by using the technique of progressive growing. As we can see, the generated images are of high quality and look very realistic. The figure has been taken from our work P-Nets [68].

Progressive GAN [23] is able to generate images of an extreme level of detail of resolutions up to 1024×1024. To achieve this, the authors in [23] proposed the gradual growing of the resolution of the images as the training progresses. As a result, the GAN first starts to generate images of e.g., resolution 4×4, then moves on to 8×8, and so on and so forth, till it reaches the desired resolution. Finally, in order to stabilise the training process and the transition between the different resolutions, the authors introduced some novel concepts such as the equalised learning rate [23]. Some samples generated by utilising the technique of progressive growing in GANs are shown in Fig. 2.12.

2.8 Conditional Generative Adversarial Networks

All of the GAN architectures and settings we have described thus far are *unsupervised*, meaning that we cannot choose for example which class of objects to generate and hence the whole generative process is random. Nonetheless, quite often we want to generate objects that belong to a certain class or objects that bear characteristics belonging to a distinct combination of classes. In order to achieve this, conditional GANs (cGANs) [34] were introduced in the

literature.

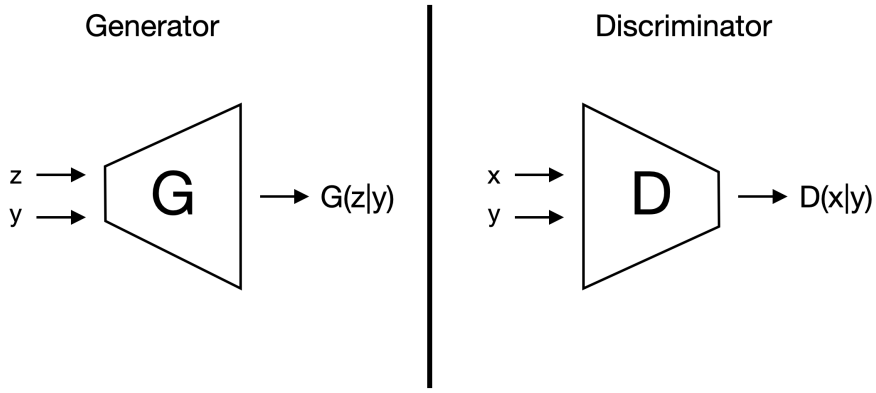


Figure 2.13: Conditional GAN structure. We denote the latent vector as $\mathbf{z}$, the label as $\mathbf{y}$ and outputs of the generator and discriminator as $G(\mathbf{z}|\mathbf{y})$ and $D(\mathbf{x}|\mathbf{y})$ (in the case where the input in the discriminator is a real datum), respectively.

In more detail, suppose we have a dataset annotated for a total of C classes (e.g., MNIST [56] dataset contains hand-written digits so the total number of classes is 10). The conditional GAN takes as input together with the random noise vector $\mathbf{z}$ the class label (which we denote as $\mathbf{y}$), which guides the generative process in order to construct a datum belonging the particular class. The same happens in the discriminator too, where the input is consisted of a real ($\mathbf{x}$) or fake ($G(\mathbf{z}|\mathbf{y})$) datum and their respective class $\mathbf{y}$. The whole process is schematically presented in Fig. 2.13, whereas some novel samples when the conditional GAN is trained on MNIST [56] dataset are shown in Fig. 2.14.

2.8.1 Image-to-Image Translation with Generative Adversarial Networks

Image translation tasks refer to applications where, given an input image, we transfer it in another setting and thus produce a new image varying in certain characteristics but without losing the global structure of the input image. For example, image translation tasks include but they are not limited to transferring black-and-white images to colour ones, converting aerial views to map views, etc.

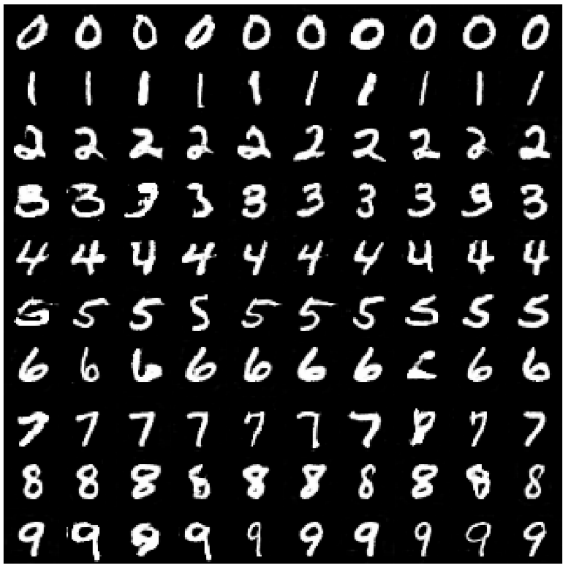


Figure 2.14: Generations using the Conditional GAN (under the P-Nets [68] variation). As can be seen, we are able to control the generative process and as a result, we produce a certain digit in every row. Moreover, the differences among the digits of the same class can be explained from the fact that we use a different latent noise vector in order to generate each one of them. The figure has been taken from our work P-Nets [68].

With the advent of GANs, the performance of image-to-image translation tasks improved significantly, owning to the ability of the GANs to generate images of very high quality. One of the most widely used (conditional) GANs to carry out image-to-image translation tasks is pix2pix [35].

More specifically, in pix2pix, the generator, which has the structure of a U-Net architecture, receives as input an image and translates it to the target domain. The discriminator then receives as input both the generated (fake) and real images that belong to the target domain and tries to discriminate the real from the fake ones. The objective that is optimised is as follows:

$$\mathcal{L} = \mathbb{E}_{\mathbf{x},\mathbf{y}} \left[\log D\left(\mathbf{x},\mathbf{y}\right)\right] + \mathbb{E}_{\mathbf{x},\mathbf{z}} \left[\log D\left(\mathbf{x}, G\left(\mathbf{x},\mathbf{z}\right)\right)\right] + \mathbb{E}_{\mathbf{x},\mathbf{z},\mathbf{y}} \left[\left\|\mathbf{y} - G\left(\mathbf{x},\mathbf{z}\right)\right\|_1\right], \tag{2.28}$$

where the $L1$ reconstruction loss is added to further assist the generator in its task.

2.9 Conclusions

In this Chapter, we presented the most important research works that are related to our publications which we detail later on. In particular, we started with the most closely related deterministic as well as probabilistic CA methodologies and we continued with the relevant DL methodologies, giving an emphasis on GANs. Despite the popularity of the techniques discussed in this Chapter, the CA methods fail to perform optimally on multi-attribute data, whereas as the DL methods do not perform well when trained on 3D data. In the Chapters that follow, we present our publications which help mitigate the aforementioned problems.

Chapter 3

AgeDB: The first manually collected, in-the-wild age database

Contents

3.1 Introduction

Adamantly, one of the most challenging and important tasks in Computer Vision throughout the years has been face recognition. This spark of interest in face recognition was firstly attributed to security reasons, since automatic facial analysis would assist security agencies in detection of passport frauds, identification of criminals or missing children, restriction of identity thefts, etc. The first algorithm for face recognition was introduced in mid 1960's [69]

and since then numerous approaches have been proposed in the literature. Before the advent of deep learning in 2006 [70], construction of algorithms that were used in facial analysis tasks required huge amount of time, domain specific knowledge and a delicate engineering process in order to transform the raw facial data in a feature space. The derived features would then be utilised to produce the final classification result [17, 18].

Nevertheless, successful implementation of face recognition applications also requires sufficiently large facial datasets that will be utilised in order to train the algorithms. For decades and prior to the proliferation of deep learning, various facial datasets were introduced in the literature. One common feature of all these datasets was that they contained images which were captured under controlled conditions (e.g., common background, controlled lighting setting, etc.). This restriction was imposed due to the fact that the feature extractors did not perform well on "in-the-wild" datasets (i.e., datasets that include images captured in uncontrolled conditions). Some of the most widely used databases which included images captured under controlled conditions were the XM2VTS database [71], the Multi-PIE database [72, 73], the AR face database [74], the Caltech faces database [75], the FERET database [76, 77, 78], the Yale face database [79].

During the last few years, an explosion in scientific research with respect to the development of deep learning architectures for face recognition has been witnessed. Deep learning comprises a set of methods that are able to automatically discover the patterns that may exist in raw data and, as a result, feature extractors which were utilised to transform the raw data are no longer required [17, 18]. This competitive advantage of deep learning methods against the conventional algorithms led to the introduction of several "in-the-wild" databases in the literature. The term "in-the-wild" is used to refer to databases that contain images which have been captured under completely uncontrolled conditions (e.g., varying backgrounds, existence of noise in the pictures, existence of occlusions in the faces depicted in various images, different types of cameras used to capture the images, etc.).

Over the past decade, various "in-the-wild" facial databases became publicly available. More specifically, in 2007 the Labeled Faces "in-the-wild" (LFW) database [80, 81] was introduced. LFW contains 13,233 images of 5,749 individuals, where 1,680 subjects are depicted in two or more images and the rest appear only in one image. In 2009, PubFig database [82] was introduced. PubFig is an "in-the-wild" database, containing 58,797 images of 200 subjects. In 2011, YouTube Faces (YTF) database [83] was introduced. YTF contains 3,425 videos of 1,595 individuals, where the average length of each video is 181.3 frames. In 2012, WDRef database [84] was introduced. WDRef contains 99,773 images of 2,995 individuals, where 2,995 subjects have 15 or more images. In 2014, the CelebFaces database [13, 14, 85] was introduced. CelebFaces contains 202,599 images of 10,177 identities (celebrities), where each identity has about 20 images. In 2014, CASIA-WebFace database [86] was introduced. CASIA-WebFace contains 494,414 images pertaining to 10,575 subjects. In 2015, VGG Face dataset [87] was introduced. VGG Face dataset contains 2.6M image of 2,622 distinct individuals. Moreover, in 2015, the IARPA Janus Benchmark A (IJB-A) [88] was introduced. IJB-A contains 5,712 images and 2,085 videos from 500 subjects. In 2015 and 2016, the MegaFace database [89, 90] was introduced and extended, respectively. MegaFace contains 4.7M images of 672K individuals. A table providing an overview of the most recently introduced aforementioned databases along with their essential statistics is presented in Table 3.1.

Table 3.1: Concise overview of the most broadly used "in-the-wild" facial datasets in the Computer Vision community since 2007 onwards.

Dataset	**Year**	**# Images**	**# Videos**	**# Subjects**	**"In-the-wild"**
LFW [80, 81]	2007	13,233	–	5,749	Yes
PubFig [82]	2009	58,797	–	200	Yes
YTF [83]	2011	–	3,425	1,595	Yes
WDRef [84]	2012	99,773	–	2,995	Yes
CelebFaces [13, 14, 85]	2014	202,599	–	10,177	Yes
CASIA-WebFace [86]	2014	494,114	–	10,575	Yes
VGG Face [87]	2015	2.6M	–	2,622	Yes
IJB-A [88]	2015	5,712	2,085	500	Yes
MegaFace [89, 90]	2016	4.7M	–	672K	Yes

Due to the recent surge of deep learning, age estimation from facial images has gradually gathered increased interest in the community. As of today, various deep learning architectures have been proposed and several databases annotated with regard to the age attribute have been made publicly available. The first database which contained images annotated with respect to the age attribute was FG-NET [91], introduced in 2002. FG-NET includes 1,002 images, captured under controlled conditions, pertaining to 82 subjects with accurate to the year age annotations. In 2006, MORPH database [92] was introduced. MORPH contains 1,724 images, pertaining to 464 subjects with accurate to the year age annotations. In 2008, the UIUC-IFP-Y Internal Aging Database [93, 94] was introduced, containing 8,000 images, captured under controlled conditions, pertaining to 1600 subjects with accurate to the year age annotations. In 2009, Gallagher group photos [95] was introduced. Gallagher group photos is an "in-the-wild" database containing 28,231 images of 5,080 subjects. As far as the annotation with the regard to the age attribute is concerned, 7 distinct age groups are utilised. Each image is annotated with a unique group identifier. In 2011, VADANA database [96] was introduced. VADANA is an "in-the-wild" database containing 2,298 images pertaining to 43 subjects. In total 4 age groups are utilised and each image is annotated with a unique group identifier. In 2014, AdienceFaces database [97, 98] was introduced, containing 26,580 "in-the-wild" images of 2,984 subjects. In total 8 age groups are utilised and each image is annotated with a unique group identifier. In 2014, Cross-Age Celebrity Dataset (CACD) [99, 100] was introduced, containing 163,446 "in-the-wild" images of 2,000 celebrities. Images are annotated with a specific age label which is semi-automatically estimated. In 2015, IMDB-WIKI database [101, 102] was introduced, containing 523,051 "in-the-wild" images pertaining to 20,284 celebrities. Images are annotated with a specific age label which is semi-automatically estimated. A table providing an overview of the aforementioned databases along with the newly introduced AgeDB database is presented in Table 3.2.

Nevertheless, despite of the increased interest in age estimation "in-the-wild" and the several

Table 3.2: Concise overview of the most broadly used age datasets in the Computer Vision community since 2002 onwards.

Dataset	Year	# Images	# Subjects	Age labels	Noise-free labels	"In-the-wild"
FG-NET [91]	2002	1,002	82	Accurate to the year	Yes	No
MORPH [92]	2006	1,724	464	Accurate to the year	Yes	No
IFP-Y [93, 94]	2008	8,000	1600	Accurate to the year	Yes	No
Gallagher [95]	2009	28,231	5,080	7 age groups	No	Yes
VADANA [96]	2011	2,298	43	4 age groups	Yes	Yes
AdienceFaces [97, 98]	2014	26,580	2,984	8 age groups	Yes	Yes
CACD [99, 100]	2014	163,446	2,000	Accurate to the year	No	Yes
IMDB-WIKI [101, 102]	2015	523,051	20,284	Accurate to the year	No	Yes
AgeDB	2017	16,488	568	Accurate to the year	Yes	Yes

databases that came into existence to tackle this task over the last years, no manually collected "in-the-wild" database with accurate to the year age annotations has been introduced in the literature. In this Chapter we present the first manually collected "in-the-wild" age database, dubbed AgeDB. AgeDB contains images of several subjects annotated with accurate to the year age labels. The fact that AgeDB is manually collected to ensure the accuracy of the age labels comes with several advantages:

- AgeDB can be used in age-invariant face verification "in-the-wild" experiments, i.e., the sensitivity in the performance of face recognition algorithms can be measured as the age gap between instances (images) of the same subject increases. Since the age labels are clean, AgeDB ensures a noise-free evaluation of the various face recognition algorithms.
- AgeDB can be used in age estimation "in-the-wild" experiments. Since the age labels are clean, AgeDB may be utilised as a benchmark database for such tasks.
- AgeDB can be utilised in face age progression "in-the-wild" experiments, since it is a manually collected database with large range of ages for each subject. This property renders AgeDB highly beneficial when training models for age progression experiments.

The structure of the Chapter is summarised as follows. In Section 2, we provide all the

necessary details pertaining to the AgeDB database. In Section 3, we present the various experiments we performed on AgeDB. More specifically, we perform: i) various age-invariant face verification "in-the-wild" experiments utilising state-of-the-art pre-trained deep networks and report their performance on AgeDB, ii) various age estimation "in-the-wild" experiments utilising state-of-the-art pre-trained deep networks and report their performance on AgeDB, iii) several face age-progression "in-the-wild" experiments and report their results on AgeDB.

3.2 The AgeDB database

In this Section we thoroughly discuss all the details pertaining to the collection of the AgeDB database as well as provide the necessary statistics related to the database. The database is publicly available at: https://ibug.doc.ic.ac.uk/resources/agedb/.

As aforementioned, AgeDB is a *manually* collected database to ensure that the age labels are clean as opposed to other age databases which have been semi-automatically collected utilising crawlers and in which the age labels may be noisy. In order to achieve noise-free annotation with respect to the age labels, we manually searched for images through Google Images and subsequently kept only images where the exact, accurate to the year age of each depicted subject is *explicitly* mentioned in the accompanied caption of the image. An indicative example of the process followed is provided in Fig. 3.1.

Moreover, AgeDB is an "in-the-wild" database, meaning it contains images captured under completely uncontrolled, real-world conditions (i.e., having different poses, containing noise, bearing various expressions, containing occlusions, etc.). This feature may prove highly advantageous, since most of the state-of-the-art deep networks are trained and evaluated in "in-the-wild" databases.

AgeDB contains $16,488$ images of various famous people, such as actors/actresses, writers, scientists, politicians, etc. Every image is annotated with respect to the identity, age and

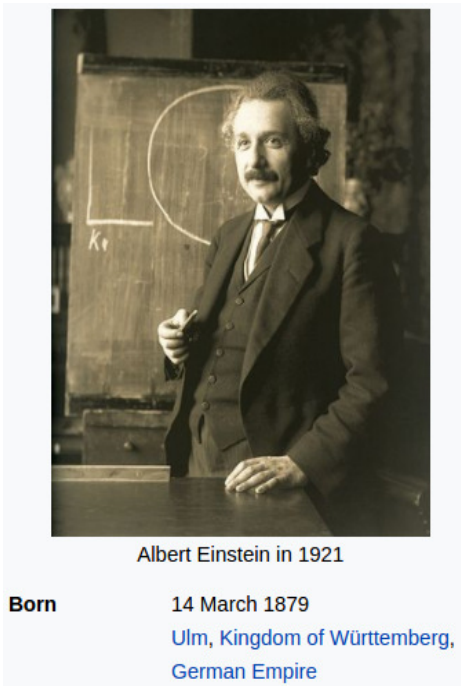


Figure 3.1: Screenshot captured from the Wikipedia lemma about Albert Einstein (http://bit.ly/AgeDB_fig1). Since the exact year the image was captured is available and also the birth year is provided, the age label can be subsequently calculated. For the image depicted in the screenshot, the age label is 42.

gender attribute. There exist a total of 568 distinct subjects. The average number of images per subject is 29. The minimum and maximum age is 1 and 101, respectively. The average age range for each subject is 50.3 years. A scatter plot depicting the age distribution of the database is presented in Fig. 3.2. Samples from the AgeDB "in-the-wild" database along with their labels are provided in Fig. 3.3.

3.3 Experiments

In this Section we present various experiments we performed on AgeDB. More specifically, we utilise state-of-the-art algorithms and conduct experiments in tasks as age-invariant face verification "in-the-wild", age estimation "in-the-wild", face age progression "in-the-wild" and subsequently show that AgeDB constitutes a proper benchmark for evaluating state-of-the-art

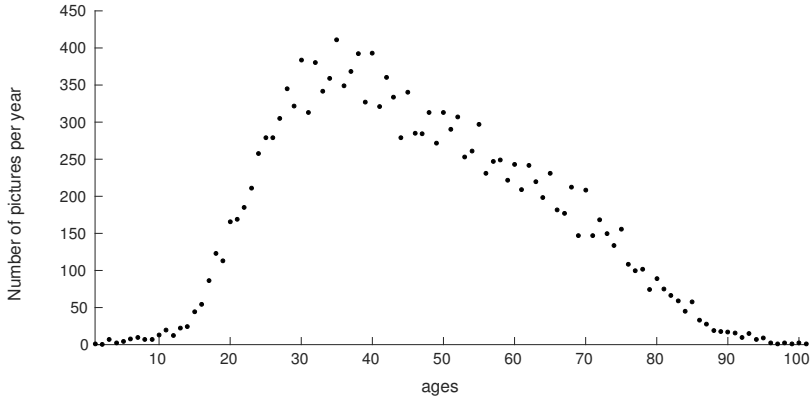


Figure 3.2: Scatter plot depicting the age distribution in the AgeDB "in-the-wild" database.

ID: *Van Damme, Jean-Claude*
Age: *27*

ID: *Douglas, Michael*
Age: *35*

ID: *Dalton, Timothy*
Age: *48*

ID: *Sinatra, Frank*
Age: *56*

ID: *Disney, Walt*
Age: *64*

Figure 3.3: Random images from the AgeDB "in-the-wild" database.

algorithms for the previously mentioned tasks.

3.3.1 Age-invariant face verification "in-the-wild"

AgeDB may be utilised for age-invariant face verification "in-the-wild" experiments since it contains multiple images of subjects depicted at different ages. To this end, in a similar fashion to the protocol that is described in the LFW [80] database, we developed four distinct age-invariant splits utilising AgeDB. More specifically, each split refers to different age gaps in terms of years (5, 10, 20, and 30 years). Each of these splits contains 300 intra-split and 300

inter-split pairs, respectively. In Table 3.4, we report the results from the utilisation of the VGG Face deep network using the aforementioned splits. In Table 3.3, we report the results from the Center Loss [103] and Marginal Loss [104] methods, again utilising the same splits. It should be noted that this series of experiments was conducted on a subset of the final version of the AgeDB, as AgeDB was further extended by the time it became publicly available. As can be seen in both experiments, AgeDB is a challenging dataset, especially as the age gap between the inter and intra splits increases. As a result, it can be widely used to benchmark algorithms tailored to the task of age-invariant face verification.

Table 3.3: Age-invariant face verification on AgeDB utilising the Centre Loss [103] and Marginal Loss [104] methods.

Age gap	Accuracy per method	
	Center Loss	Marginal Loss
5	0.959	0.981
10	0.951	0.979
20	0.931	0.971
30	0.907	0.957

Table 3.4: Age-invariant face verification on AgeDB utilising the VGG Face [87] deep network.

Age gap	Accuracy per layer			
	33-rd	34-th	35-th	36-th
5	0.912	0.931	0.911	**0.934**
10	0.900	**0.922**	0.902	0.914
20	0.879	**0.891**	0.879	0.888
30	0.839	**0.851**	0.842	0.840

3.3.2 Age estimation "in-the-wild"

Age estimation "in-the-wild" is a challenging problem in Computer Vision and has recently gained huge interest in the community, mainly due to the increased penetration of deep learning techniques in the literature. The challenging nature of this task is primarily attributed to the

fact that the databases which are publicly available and utilised for age estimation have been semi-automatically collected and thus contain age annotations that are noisy. As a result, age prediction based on such databases cannot be accurate, since the algorithms are trained on data where the labels in the first place are not accurate. To overcome the said disadvantage, we introduce the AgeDB "in-the-wild" database, the first manually collected age database.

The state-of-the-art publicly available pre-trained deep network for age estimation "in-the-wild" is DEX (Deep EXpectation of apparent age from a single image) [101, 102], winner of the LAP challenge 2015 on apparent age estimation [101]. We hence utilised the publicly available DEX pre-trained deep network [101] and performed age estimation in the totality of pictures included on AgeDB.

Preprocessing

In order to feed the images of AgeDB in the pre-trained network, we followed the preprocessing process described in [101]: We firstly employed the Mathias et al. face detection algorithm [105] and then used an extra 40% margin on the initial bounding box boundary. Moreover, we discarded the images in which the face detector was not able to extract a proper bounding box containing the face. We then used the cropped images as input in the pre-trained deep network.

Age estimation

As mentioned in [101], the output layer of the deep network corresponds to a 101 dimensional vector $\mathbf{v}$, representing softmax output probabilities, one for each age between 0 and 100 included. The final age estimation is given by the softmax expected value, i.e.:

$$\mathbb{E}[\mathbf{v}] = \sum_{n=0}^{100} y_n \cdot v_n, \tag{3.1}$$

where y_i are the years corresponding to the n-th class. A graphical overview of the processed followed in provided in Fig. 3.4.

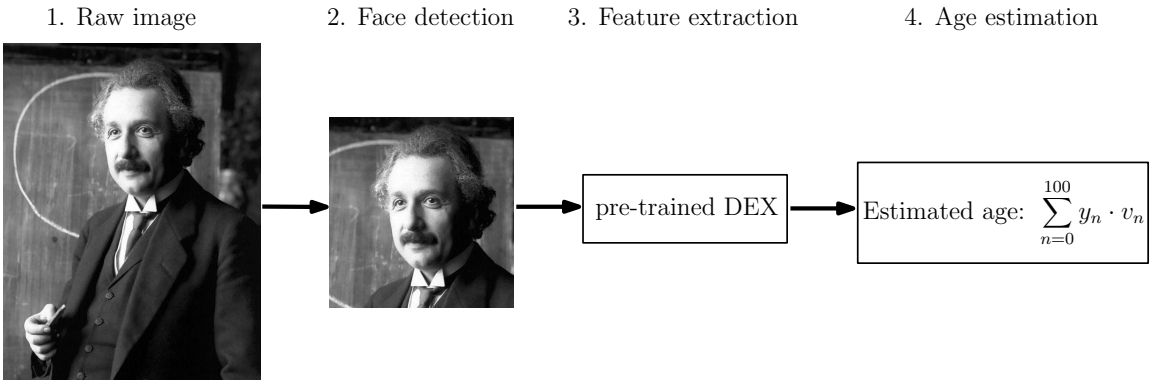


Figure 3.4: Visualisation of the complete process for age estimation utilising the pre-trained DEX deep network [101].

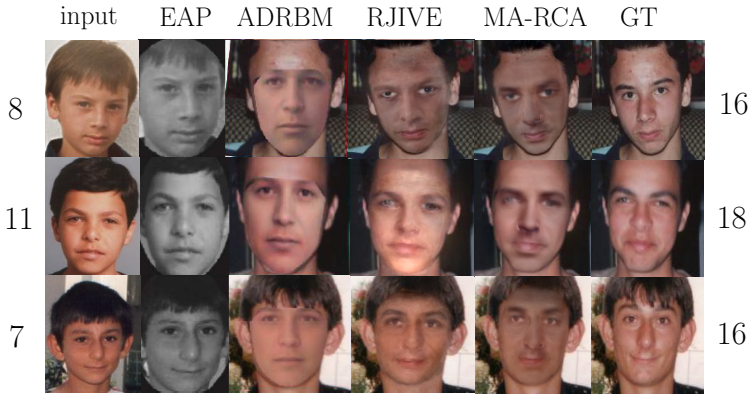


Figure 3.5: Age-progression results produced by EAP, ADRBM, RJIVE and MARCA method on images of the FG-Net database [91]. MARCA and RJIVE are trained on AgeDB [16].

Evaluation

As evaluation protocol we used the standard Mean Absolute Error (MAE), which is defined as the average of the absolute errors between the estimated age and the ground truth age. For the AgeDB "in-the-wild" database, the MAE for the pre-trained DEX deep network [101] is 13.1 years.

3.3.3 Face age progression "in-the-wild"

Face age-progression is an image translation task where, given an image depicting a subject at a specific age, we try to alter the appearance of the subject so that they look younger or older.

In order to train algorithms that can successfully carry out this task, databases annotated with respect to the age attribute should be used. Nevertheless, most of the age databases that have been made publicly available are semi-automatically collected and annotated (e.g., using web crawlers) and thus contain noisy age labels. As a result, the performance of the algorithms when trained on such databases can be negatively affected by the errors in the age labels. AgeDB overcomes the aforementioned shortcomings. To demonstrate this, we carry out age-progression experiments utilising various age-progression algorithms and show that when the AgeDB dataset is used for training, results can be considerably improved.

In particular, for the age-progression experiments we utilised the following algorithms: Illumination Aware Age Progression (IAAP) method [106], Coupled Dictionary Learning (CDL) method [107], Deep Aging with Restricted Boltzmann Machines (DARB) method [108], CG [109], Recurrent Face Aging (RFA) method [110], Robust Joint and Individual Variance Explained (RJIVE) [111] and Multi-Attribute Robust Component Analysis (MARCA) [10]. RJIVE and MARCA were trained utilising AgeDB as follows: we split AgeDB in 10 different age-groups which include the following ranges: $0-3, 4-7, 8-15, 16-20, 21-30, 31-40, 41-50, 51-60, 61-70$ and $71-100$. Then, we use these age-groups to carry out the training process and thus extract the relevant age-group subspaces after the training is complete. Finally, depending on which age-group we want to transfer the subject depicted in a test image, we make use of the relevant learnt subspace. In Fig. 3.5, progressed images produced by the compared methods are depicted. As can be seen, RJIVE and MARCA outperform the rest of the methods, deeming the advantages of utilising the AgeDB dataset apparent. Note that all progressed faces have been fused with the actual ones to retain details such as the image background. More information about MARCA can be found in Chapter 4.

3.4 Conclusion

In this Chapter we introduced the AgeDB database, the first *manually* collected, "in-the-wild" age database, which contains noise-free identity, gender and accurate to the year age labels. Moreover, we utilised AgeDB along with state-of-the-art algorithms and performed a series of experiments in tasks such as age-invariant face verification "in-the-wild", age estimation "in-the-wild", face age progression "in-the-wild". Finally, we showed that AgeDB can be utilised as a benchmark for evaluating state-of-the-art algorithms that aim to tackle the aforementioned tasks.

Chapter 4

Multi-Attribute Robust Component Analysis for Facial UV Maps

Contents

4.1 Introduction

Significant progress has been observed during the past years in the field of sparse and dense 3D face alignment [112, 113, 114, 115, 116]. Recent developments include the utilisation of Deep Neural Networks (DNNs) for estimation of 3D facial structure, as well as a methodology for fitting a 3D Morphable Model (3DMM) in "in-the-wild" images [116]. Additionally, several benchmarks for training sparse 3D face alignment models have been recently developed [114, 116]. The utilisation of these methods introduces new challenges and opportunities as far as facial texture is concerned.

In particular, by sampling over the fitted image, a 2D UV map of the facial texture can be constructed. A UV texture map is the main way to store the texture of a 3D model as a simple 2D image in graphics applications. In more detail, it is a 2D representation of the texture of a 3D model which is precisely connected with the object's vertices through a set of coordinates that map each vertex to a specific location of the UV map. The mapping coordinates naturally occur by unwrapping the 3D model and scaling it to the UV texture map's space [117]. In this work we employ UV texture maps that are precisely mapped to 3D models of human faces and particularly to instances of the LSFM face model [118].

A facial UV map may contain a considerable amount of missing data (pixels) due to factors such as self-occlusion. Nevertheless, it does not suffer from warping effects, in contrast to the facial images produced by a 2D face alignment algorithm, as can be clearly seen in Fig. 5.2. Utilising facial UV maps for the discovery of latent components suitable for specific tasks requires the design of statistical component analysis methods that (a) can appropriately handle missing values, (b) can alleviate problems arising from gross errors, and (c) exploit any existing labels/attributes that are available. To tackle the aforementioned issues, it is natural to adopt techniques from the family of robust component analysis.

a) b) c)

Figure 4.1: Original facial image is depicted in a). The facial image produced by a 2D face alignment algorithm is presented in b). The UV map derived by fitting a 3DMM is depicted in c).

In the past years, significant research has been conducted in terms of formulating robust component analysis techniques. Arguably, the most prominent example lies in the Robust PCA (RPCA) algorithm [33], that has also been extended for handling missing values in [119]. The RPCA algorithm with missing values has been recently proven extremely useful towards the extraction of a low-rank subspace of facial UV textures that is free of gross errors, thus deeming it extremely useful for the fitting of 3DMMs "in-the-wild" [116]. Nevertheless, RPCA is an unsupervised component analysis technique, and hence does not take into account the various attributes/annotations that may be present in the data at hand.

Different variations of RPCA [33] have been introduced in the literature, such as the Robust Principal Component Analysis with Non-Greedy $l1$-Norm Maximisation (RPCA-L1) [46] and the Optimal Mean Robust Principal Component Analysis (OM-RPCA) [120], both of which propose different methods for solving the RPCA optimisation problem ([33]). In RPCA-L1 [46], the authors first propose an efficient optimisation algorithm to solve a general $l1$-norm maximisation problem and then employ it to formulate a non-greedy solution of the $l1$-norm RPCA problem. OM-RPCA [120] solves the traditional $l1$-norm RPCA optimisation problem while simultaneously updating the mean component of the data matrix at each iteration of the algorithm, aiming to achieve better separation of the data. Similarly to the common RPCA algorithm [33], none of the two aforementioned methods are designed to exploit any available

labels or annotations of the data.

Other recent robust component analysis methods include Robust Correlated and Individual Component Analysis (RCICA) [121], as well as Robust Joint and Individual Variance Explained (RJIVE) [122]. RCICA robustly recovers both the correlated and individual low-rank components of two views of noisy data, and can therefore be interpreted as a robust extension of Canonical Correlation Analysis (CCA) [123]. Nevertheless, RCICA is not designed to utilise labels or any available annotations. RJIVE further extends RCICA by extracting low-rank subspaces from multiple-views similarly to RCICA in the presence of a *single* attribute only (e.g., if the data at hand are annotated for the attribute *age*, then they may be split in different age-groups and each age-group can be considered as a different view). As a result, data that is annotated in terms of multiple attributes (such as *identity* and *age*) cannot be fully exploited in RJIVE.

To alleviate the shortcomings of the previously mentioned methods, in this Chapter we introduce Multi-Attribute Robust Component Analysis, dubbed MARCA, which takes a much more natural approach to the problem of facial UV analysis, since it is able to inherently incorporate the existence of multiple attributes at hand during the training as well as the testing processes. In this way, as we also show in the experiments, MARCA is be able to extract more meaningful subspaces compared to the aforementioned methods. This can lead to better performance in tasks such as image denoising and translation. The rest of the Chapter is organised as follows. In Section 4.2 we provide the mathematical formulation of MARCA and present all the necessary optimisation algorithms. In Section 4.3 we run a series of experiments and demonstrate the merits of MARCA against other state-of-the-art algorithms.

(a) (b) (c) (d)

Figure 4.2: Extraction of a UV texture map with missing values from a 2D image using a face 3DMM. Having recovered the (b) 3D shape of a face by fitting a 3DMM into (a) an input image we can (c) project the reconstructed 3D shape on it and then compute the UV texture map along with the occluded parts.

4.2 Multi-Attribute Robust Component Analysis

4.2.1 Preliminaries

Prior to delving into the model, a few explanations regarding the notations used throughout the Chapter are provided. Lower-case letters, e.g., x, denote scalars, lower-case (upper-case) bold letters denote vectors (matrices), e.g., $\mathbf{x}$ ($\mathbf{X}$). Moreover, L_1 (L_2) vector norm is defined as $\|\mathbf{x}\|_1 \doteq \sum_i |x_i|$ ($\|\mathbf{x}\|_2 \doteq \sqrt{\sum_i x_i^2}$). Similarly, $L_{1,1} \equiv L_1$ ($L_{2,2} \equiv L_F$) matrix norm is defined as $\|\mathbf{X}\|_1 \doteq \sum_{i,j} |x_{ij}|$ ($\|\mathbf{X}\|_F \doteq \sqrt{\sum_{i,j} x_{ij}^2}$). The nuclear norm of a matrix $\mathbf{X}$, i.e., the sum of its singular values, is defined as $\|\mathbf{X}\|_*$. The Hadamard, i.e., element-wise, product of two matrices $\mathbf{X}$ and $\mathbf{Y}$ is denoted as $\mathbf{X} \odot \mathbf{Y}$. Finally, we provide the following operator definitions which will be utilised in the mathematical derivations required for MARCA.

- Procrustes operator: $\mathcal{Q}(\mathbf{X}) \doteq \mathbf{U}\mathbf{V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are given by the rank-r Singular Value Decomposition (SVD) of $\mathbf{X}$, i.e., $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$.

- Shrinkage operator: $\mathcal{S}_\tau(\sigma) \doteq \operatorname{sgn}(\sigma) \max(|\sigma| - \tau, 0)$

- Singular Value Thresholding (SVT) operator: $\mathcal{D}_\tau(\mathbf{X}) \doteq \mathbf{U}\operatorname{diag}(\mathcal{S}_\tau(\mathbf{d}))\mathbf{V}^T$, where $\mathbf{X} = \mathbf{U}\operatorname{diag}(\mathbf{d})\mathbf{V}^T$ is the SVD of $\mathbf{X}$.

4.2.2 Problem formulation

Without any loss of generality, suppose that the incomplete, contaminated with gross errors UV maps are annotated for J attributes (e.g., *identity*, *age*, etc.), where each attribute may have $M_i, \forall i \in \{1, \ldots, J\}$, different instantiations (e.g., attribute *identity* may have the instantiation *Frank Sinatra*, *Albert Einstein*, etc.). Moreover, assume that there is a total of N samples in the training set. Aim of MARCA is to robustly extract J joint components corresponding to the available attributes during training, an individual component which captures the rest data information that cannot be explained by the J components and a component which captures the gross but sparse errors. Let training data be concatenated in a column-wise manner, i.e., $\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \ldots \mathbf{x}_N \end{bmatrix}$, where $\mathbf{x}_i \in \mathbb{R}^{F \times 1}, i \in \{1, \ldots, N\}$, is a vectorised form of a facial UV map. Then MARCA admits the following decomposition.

$$\mathbf{X} = \sum_{i=1}^{J} \mathbf{S}_i + \mathbf{G} + \mathbf{E}, \tag{4.1}$$

where $\mathbf{S}_i, i \in \{1, \ldots, J\}$, are the J shared components for every attribute, $\mathbf{G}$ is the individual component and $\mathbf{E}$ is the error component.

Nevertheless, $\mathbf{S}_i, i \in \{1, \ldots, J\}$, must have a specific low-rank structure which accounts for the different instantiations of each attribute. That is, every attribute should be rendered by a *base* and subsequently every corresponding instantiation be rendered by a *selector* on that base. Therefore, (4.1) is re-formulated as follows.

$$\mathbf{X} = \sum_{i=1}^{J} \mathbf{F}_i \mathbf{H}_i + \mathbf{G} + \mathbf{E}, \tag{4.2}$$

where $\mathbf{F}_i \in \mathbb{R}^{F \times M_i}, i \in \{1, \ldots, J\}$, are the *bases* that render each attribute and $\mathbf{H}_i \in \mathbb{R}^{M_i \times N}, i \in \{1, \ldots, J\}$, are comprised of the *shared selectors*, i.e., $\mathbf{H}_i \doteq \begin{bmatrix} \mathbf{h}_{i,a} \ldots \mathbf{h}_{i,b} \end{bmatrix}$ (such that $a, b \in \{1, \ldots, M_i\}$), which render a specific instantiation for an attribute (e.g., assuming that base $\mathbf{F}_i$ renders attribute *identity*, then $\mathbf{h}_{M_i}$ would render a particular instantiation of this attribute, e.g., *Albert Einstein*). It should be noted that data which bear the same instantiation (e.g., M_i) for a particular attribute (e.g., attribute i) have the same selector $\mathbf{h}_{i,M_i}$

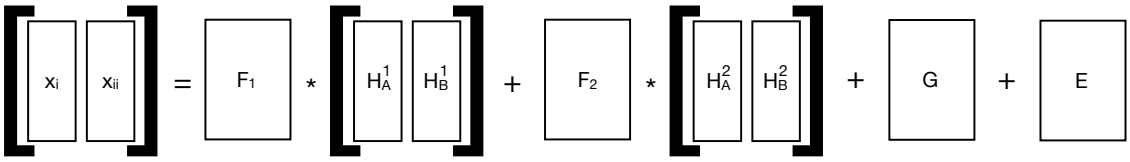


Figure 4.3: Decomposing the data in different subspaces in MARCA in the simple case where only two data $\mathbf{x}_i$ and $\mathbf{x}_{ii}$, two attributes and two labels per attribute are available. Subspaces $\mathbf{F}_1$ and $\mathbf{F}_2$ describe each attribute, and the selectors $\mathbf{H}_A^1$, $\mathbf{H}_B^1$ and $\mathbf{H}_A^2$, $\mathbf{H}_B^2$ describe the instantiations/labels of each attribute, respectively. Subspace $\mathbf{G}$ captures the rest of the information that cannot be explained by the attributes and finally subspace $\mathbf{E}$ captures the gross but sparse error.

(for instance, multiple data with instantiation *Albert Einstein* will all have the selector $\mathbf{h}_{i,M_i}$). A visualisation of the above decomposition is presented in Fig. 4.3. Furthermore, low-rank base $\mathbf{G} \in \mathbb{R}^{F \times N}$, renders the individual variation for all of the images in the training set that cannot be explained by the existing attributes. Finally, $\mathbf{E} \in \mathbb{R}^{F \times N}$ encapsulates gross errors (such as occlusions, pixel corruptions, etc.) for all of the data samples in the training set.

In order to recover components $\{\mathbf{F}_i \mathbf{H}_i\}_{i=1}^{J}$ and $\mathbf{G}$ which are as informative as possible, the error term which accounts for the existence of gross but sparse errors in the visible parts of the UVs has to be minimised. This is equivalent to minimising the L_0 norm of the error term for the visible parts of the UV maps. However, to avoid the NP-hardness of the L_0 norm minimisation we adopt the L_1 norm as the tightest convex surrogate [124].

The problem is then formulated as follows.

$$\begin{aligned} &\min_{\theta} \|\mathbf{W} \odot \mathbf{E}\|_1 , \\ \text{s.t. } &\mathbf{X} = \sum_{i=1}^{J} \mathbf{F}_i \mathbf{H}_i + \mathbf{G} + \mathbf{E} \\ &\left\{\mathbf{F}_i^T \mathbf{F}_i = \mathbf{I}\right\}_{i=1}^{J}, \operatorname{rank}(\mathbf{G}) = R, \end{aligned} \tag{4.3}$$

where $\theta = \{\mathbf{F}_i, \mathbf{H}_i, \mathbf{G}\}, i \in \{1, \ldots, J\}$ and $R, R < \min(F, N)$, is a hyper-parameter. Moreover, $\mathbf{W} \doteq \begin{bmatrix} \mathbf{w}_1 \ldots \mathbf{w}_N \end{bmatrix}$, where $\mathbf{w}_i \in \{0,1\}^{F \times 1}, i \{1, \ldots, N\}$, is the corresponding vectorised occlu-

sion mask for each UV map $\mathbf{x}_i$. The visible (missing) pixels for each UV map correspond to ones (zeros) in each matching occlusion mask. Orthonormalisation constraints on the bases $\{\mathbf{F}_i\}_{i=1}^{J}$ facilitate the recovery of unique and identifiable selectors. Because of the fact that R is a hyper-parameter, it requires a large number of experiments to estimate the optimal rank for $\mathbf{G}$. Since R is upper bounded, the following relaxed decomposition can be used to automatically recover the optimal rank for $\mathbf{G}$.

$$\begin{aligned} &\min_{\theta} \lambda \left\|\mathbf{W} \odot \mathbf{E}\right\|_1 + \left\|\mathbf{G}\right\|_*, \\ \text{s.t. } &\mathbf{X} = \sum_{i=1}^{J} \mathbf{F}_i \mathbf{H}_i + \mathbf{G} + \mathbf{E}, \\ &\left\{\mathbf{F}_i^T \mathbf{F}_i = \mathbf{I}\right\}_{i=1}^{J}, \end{aligned} \tag{4.4}$$

where the nuclear norm of $\mathbf{G}$ is introduced as a convex surrogate of the rank function [33] and $\lambda > 0$ is a regulariser.

Replacing the parameter R with the regulariser λ makes the problem much easier to address and in much less time [33, 122]. Regulariser λ can be selected by using the formula $\lambda = 1/\sqrt{max(F, N)}$ (where F, N correspond to the rows and columns of the data matrix, respectively), as shown in [33, 122].

4.2.3 Mathematical derivations

Because problem (4.4) is separable, we adopt an alternating optimisation scheme to find the updates for every parameter. The corresponding partially Augmented Lagrangian for (4.4) may then be written as

$$\begin{aligned} \mathcal{L}(\theta) = \lambda \left\|\mathbf{W} \odot \mathbf{E}\right\|_1 + \left\|\mathbf{G}\right\|_* - \frac{1}{2\mu} \left\|\mathbf{\Lambda}\right\|_F^2 + \frac{\mu}{2} \left\| \mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i \mathbf{H}_i - \mathbf{G} - \mathbf{E} + \frac{\mathbf{\Lambda}}{\mu} \right\|_F^2, \\ \text{s.t. } \left\{\mathbf{F}_i^T \mathbf{F}_i = \mathbf{I}\right\}_{i=1}^{J}, \end{aligned} \tag{4.5}$$

where $\theta = \{\mathbf{F}_i, \mathbf{H}_i, \mathbf{G}, \mathbf{\Lambda}\}, i \in \{1, \ldots, J\}$. Problem (4.5) is minimised by employing the Alternating Direction Method of Multipliers (ADMM) [45, 44]. The algorithm for solving

(4.5) is presented in Algorithm 6. The algorithm terminates when the iterations reach a predefined maximum value or a convergence criterion is met. The convergence criterion is met when the normalised reconstruction error is less than a predefined threshold ϵ (i.e., $\left\|\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t]\mathbf{H}_i[t] - \mathbf{G}[t] - \mathbf{W} \odot \mathbf{E}[t]\right\|_F / \|\mathbf{X}\|_F$). The ADMM iteration reads as follows.

Update the primal variables:

For obtaining $\mathbf{H}_i, i \in \{1, \ldots, J\}$, where, as previously mentioned, $\mathbf{H}_i = \begin{bmatrix} \mathbf{h}_{i,a} \ldots \mathbf{h}_{i,b} \end{bmatrix}$, we need to solve individually for every $\mathbf{h}_{i,j}, j \in \{1, \ldots, M_i\}$. Based on (4.5), the solution is given by minimising

$$\mathbf{h}_{i,j}[t+1] = \underset{\mathbf{h}_{i,j}[t]}{\operatorname{argmin}} \left\| \mathbf{X} - \sum_{k=1}^{J} \mathbf{F}_k[t]\mathbf{H}_k[t] - \mathbf{G}[t] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]} \right\|_F^2. \tag{4.6}$$

Problem (4.6) admits a closed-form solution, which is

$$\mathbf{h}_{i,j}[t+1] = \frac{1}{N_{i,j}} \left(\mathbf{F}_i^{i,j}[t]\right)^T \left(\mathbf{X}^{i,j} - \sum_{k=1, k\neq i}^{J} \mathbf{F}_k^{i,j}[t]\mathbf{H}_k^{i,j}[t] - \mathbf{G}^{i,j}[t] - \mathbf{E}^{i,j}[t] + \frac{\mathbf{\Lambda}^{i,j}[t]}{\mu[t]} \right) \cdot \mathbf{1}, \tag{4.7}$$

where superscript $\{i, j\}$ means that only the $N_{i,j}$ columns corresponding each time to the j-th instantiation of the i-th attribute are considered (e.g., columns corresponding to data annotated for attribute *identity* and instantiation *Albert Einstein*) and $\mathbf{1}$ is a column vector of $N_{i,j}$ ones.

For deriving subspace $\mathbf{F}_i, i \in \{1, \ldots, J\}$, the following needs to be solved.

$$\mathbf{F}_i[t+1] = \underset{\mathbf{F}[t]}{\operatorname{argmin}} \left\| \mathbf{X} - \sum_{j=1}^{J} \mathbf{F}_j[t]\mathbf{H}_j[t+1] - \mathbf{G}[t] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]} \right\|_F^2, \tag{4.8}$$
$$\text{s.t. } \left\{\mathbf{F}_i^T \mathbf{F}_i = \mathbf{I}\right\}_{i=1}^{J}.$$

In order to solve (4.8), we rely on the Procrustes Operator $\mathcal{Q}$ and the Lemma introduced next.

Lemma: The constraint minimisation problem

$$\mathbf{\Omega}^* = \underset{\mathbf{\Omega}}{\operatorname{argmin}} \|\mathbf{\Omega}\mathbf{A} - \mathbf{B}\|_F^2 \tag{4.9}$$
$$\text{s.t. } \mathbf{\Omega}^T\mathbf{\Omega} = \mathbf{I},$$

has a closed-form solution [125] of the form $\mathbf{\Omega}^* = \mathcal{Q}\left(\mathbf{B}\mathbf{A}^T\right)$. As a result, the solution for (4.8), taking into account (4.9), is

$$\mathbf{F}_i[t+1] = \mathcal{Q}\left[\left(\mathbf{X} - \sum_{j=1, j\neq i}^{J} \mathbf{F}_j[t]\mathbf{H}_j[t+1] - \mathbf{G}[t] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right) \cdot \mathbf{H}_i[t+1]^T\right]. \tag{4.10}$$

For obtaining subspace $\mathbf{G}$, the following needs to be solved.

$$\mathbf{G}[t+1] = \underset{\mathbf{G}[t]}{\operatorname{argmin}} \left[\frac{\mu[t]}{2}\left\|\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right\|_F^2 + \|\mathbf{G}[t]\|_*\right]. \tag{4.11}$$

Problem (4.11) is solved utilising the SVT operator $\mathcal{D}$. The solution is

$$\mathbf{G}[t+1] = \mathcal{D}_{\frac{1}{\mu[t]}}\left[\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right]. \tag{4.12}$$

For obtaining $\mathbf{E}$, the following problem needs to solved.

$$\begin{aligned}\mathbf{E}[t+1] = \underset{\mathbf{E}[t]}{\operatorname{argmin}} \Bigg[\lambda \|\mathbf{W} \odot \mathbf{E}[t]\|_1 + \frac{\mu[t]}{2}\Bigg\|\mathbf{X} - \\ - \sum_{i=1}^{J} \mathbf{F}_i[t+1] \cdot \mathbf{H}_i[t+1] - \mathbf{G}[t+1] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\Bigg\|_F^2\Bigg]\end{aligned} \tag{4.13}$$

Problem (4.13) is solved utilising the Shrinkage operator $\mathcal{S}$. The solution is

$$\mathbf{E}[t+1] = \mathbf{W} \odot \mathbf{Y} + \overline{\mathbf{W}} \odot \left[\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t+1] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right], \tag{4.14}$$

where

$$\mathbf{Y} = \mathcal{S}_{\frac{\lambda}{\mu[t]}}\left[\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t+1] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right] \tag{4.15}$$

and $\overline{\mathbf{W}}$ is the complement of $\mathbf{W}$.

Update the Lagrange multiplier and μ_t parameter:

$$\mathbf{\Lambda}[t+1] = \mathbf{\Lambda}[t] + \mu[t]\left(\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t+1] - \mathbf{E}[t+1]\right), \tag{4.16}$$

$$\mu[t+1] = \min\left(\rho\mu[t], \mu_{\max}\right). \tag{4.17}$$

Regarding the theoretical convergence of the ADMM algorithm presented previously, there is no proof when ADMM is utilised in settings with more than two blocks of variables. Nevertheless, ADMM provides good results in non-linear optimisation problems [126]. Experimental evaluation of MARCA in a number of different tasks on various facial datasets admits that the derived solutions constitute a good approximation. It should be noted that despite using convex surrogates for the L_0 norm and the rank function, the problem is still non-convex due to the product terms of the MARCA formulation. Non-convex relaxation techniques could be utilised in a future work.

The time complexity for MARCA at each iteration of Algorithm 1 is $\sum_{i=1}^{J} O\left(\max\left(M_i F^2\right)\right) + O\left(\left(\max\left(NF^2, F^2N\right)\right)\right) = O\left(\max\left(FN^2, F^2N\right)\right)$, due to SVD applied on both the shared J components and the individual one and since $M_i < \min(F, N), \forall i \in \{1, \ldots, J\}$. The memory complexity of MARCA is $O\left(FN\right)$, due to SVD.

As mentioned previously, the type of noise which is mostly prevalent in facial UV maps is sparse, gross, non-Gaussian noise, due to occlusions, subjects wearing eye-glasses, pixel corruptions, etc. This is why MARCA is formulated to explicitly account for the sparse noise, as shown in (4.4). Furthermore, by increasing the regulariser λ (i.e., by reducing the effect of the sparse error term in (4.4)), MARCA will be able to handle data contaminated with both Gaussian and sparse noise. Finally, MARCA is able to handle data solely contaminated with Gaussian noise by vanishing the sparse error term, i.e., by setting $\lambda \to \infty$. This is further corroborated in Section 4.3.1 and Section 4.3.2.

4.2.4 Reconstruction of a test image

After the optimal bases and the selectors have been recovered as described in Section 4.2.3, they may be utilised in order to recover the shared and individual components of a test image. Then, the said components can be utilised in experiments such as UV denoising and age progression "in-the-wild", as demonstrated in Section 4.3.

Algorithm 6 ADMM solver for problem (4.5)

Input: $\mathbf{X}$, where $\mathbf{X}$ is the set of the training data, $M_i, i \in \{1, \ldots, J\}$, where M_i is number of instantiations for every attribute i, $\mathbf{W}$, where $\mathbf{W}$ is the set of occlusion masks corresponding to the training data and $\mathbf{L}$, where $\mathbf{L}$ is a one hot representation matrix of the instantiations assigned to each image over all the available attributes. **Initialisations:** $t = 0, \epsilon = 1e-8$, maximum number of iterations $t_{\max}, \mu[0] = \frac{25}{\|\mathbf{X}\|}$, $\lambda = 1/\sqrt{max(F,N)}$ $\{\mathbf{F}_i[0], \mathbf{H}_i[0], \mathbf{G}[0], \mathbf{\Lambda}[0]\} = \mathbf{0}, i \in \{1, \ldots, J\}, \rho = 1.2, \mu_{\max} = 10^7, \{\mathbf{F}_i[0]\}_{i=1}^J =$ random initialisations so that $\mathbf{F}_i[0], i \in \{1, \ldots, J\}$, are orthonormal. **Output:** $\{\mathbf{F}_i[T], \mathbf{H}_i[T]\}_{i=1}^J, \mathbf{G}[T], \mathbf{E}[T]$.

while *not converged* **do**

  for $i = 1 : J$ **do**

    Update $\mathbf{H}_i$:

    for $j = 1 : M_i$ **do**

$$\mathbf{h}_{i,j}[t+1] = \frac{1}{N_{i,j}} \left(\mathbf{F}_i^{i,j}[t]\right)^T \left(\mathbf{X}^{i,j} - \sum_{k=1, k\neq i}^{J} \mathbf{F}_k[t]\mathbf{H}_k^{i,j}[t] - \mathbf{G}^{i,j}[t] - \mathbf{E}^{i,j}[t] + \frac{\mathbf{\Lambda}^{i,j}[t]}{\mu[t]}\right) \cdot \mathbf{1}$$

    end

$$\mathbf{H}_i[t+1] = \left[\underbrace{\mathbf{h}_{i,1}[t+1] \ldots \mathbf{h}_{i,a}[t+1]}_{N_{i,a} \text{ times}} \ldots \underbrace{\mathbf{h}_{i,b}[t+1] \ldots \mathbf{h}_{i,b}[t+1]}_{N_{i,b} \text{ times}}\right] \text{ s.t. } a, b \in \{1, \ldots, M_i\}$$

  end

  Update $\mathbf{F}$:

  for $i = 1 : J$ **do**

$$\mathbf{F}_i[t+1] = \mathcal{Q}\left[\left(\mathbf{X} - \sum_{j=1, j\neq i}^{J} \mathbf{F}_i[t]\mathbf{H}_i[t+1] - \mathbf{G}[t] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right) \mathbf{H}_i^T[t+1]\right]$$

  end

  Update $\mathbf{G}$:

$$\mathbf{G}[t+1] = \mathcal{D}_{1/\mu[t]}\left[\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{E}[t] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right]$$

  Update $\mathbf{E}$:

$$\mathbf{E}[t+1] = \mathcal{S}_{\frac{\lambda}{\mu[t]}}\left[\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t+1] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right]$$

$$\mathbf{E}[t+1] = \mathbf{W} \odot \mathbf{E}[t+1] + \overline{\mathbf{W}} \odot \left[\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t+1] + \frac{\mathbf{\Lambda}[t]}{\mu[t]}\right]$$

  Update $\mathbf{\Lambda}$:

$$\mathbf{\Lambda}[t+1] = \mathbf{\Lambda}[t] + \mu[t]\left(\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i[t+1]\mathbf{H}_i[t+1] - \mathbf{G}[t+1] - \mathbf{E}[t+1]\right)$$

  Update μ:

  $\mu[t+1] = \min\left(\rho\mu[t], \mu_{\max}\right)$

end

Without any loss of generality, assume a test UV map $\mathbf{y}$, which may be decomposed in the shared and individual components as follows.

$$\mathbf{y} = \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i + \mathbf{K}\hat{\mathbf{w}} + \hat{\epsilon}, \tag{4.18}$$

where $\mathbf{K}$ is the linear span of $\mathbf{G}$, given by applying the rank-r SVD on $\mathbf{G}$. In the most general case, optimal selectors $\left\{\hat{\mathbf{h}}_i\right\} =_{i=1}^{J}$ and $\hat{\mathbf{w}}$ must be extracted by minimising the sparse error term $\hat{\boldsymbol{\epsilon}}$ corresponding to the visible part of $\mathbf{y}$, for already recovered $\{\mathbf{F}_i\}_{i=1}^{J}$ and $\mathbf{G}$. That is, the following needs to be solved.

$$\begin{aligned} &\min_{\theta} \|\mathbf{w}_y \odot \hat{\boldsymbol{\epsilon}}\|_1 \\ \text{s.t.} \quad &\mathbf{y} = \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i + \mathbf{K}\hat{\mathbf{w}} + \hat{\boldsymbol{\epsilon}}, \\ &\hat{\mathbf{y}} = \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i + \mathbf{K}\hat{\mathbf{w}}, \end{aligned} \tag{4.19}$$

where $\mathbf{w}_y$ is the occlusion mask corresponding to the test UV map $\mathbf{y}$ and $\hat{\mathbf{y}}$ is the reconstructed, error-free facial UV map. In the case where, e.g., transfer of a test image to a specific age is required, the selector corresponding to the specific age will be fixed (i.e., the corresponding optimal selector found during the training process in Section 4.2.3 is utilised). Because problem (4.19) is separable, we adopt an alternating optimisation scheme to find the updates for every parameter. The corresponding partially Augmented Lagrangian for (4.4) may then be written as

$$\mathcal{L}(\theta) = \lambda \|\mathbf{w}_y \odot \hat{\boldsymbol{\epsilon}}\|_1 - \frac{1}{2\mu}\|\boldsymbol{\Lambda}\|_F^2 + \frac{\mu}{2}\left\|\mathbf{X} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i - \mathbf{K}\hat{\mathbf{w}} - \hat{\boldsymbol{\epsilon}} + \frac{\boldsymbol{\Lambda}}{\mu}\right\|_F^2, \tag{4.20}$$

where $\theta = \left\{\hat{\mathbf{h}}_i, \hat{\mathbf{w}}, \boldsymbol{\Lambda}\right\}, i \in \{1, \ldots, J\}$. Problem (4.20) is minimised by employing the ADMM. The algorithm for solving (4.20) is presented in Algorithm 7. The algorithm terminates when the iterations reach a predefined max value or a convergence criterion is met. The convergence criterion is met when the normalised reconstruction error is less than a predefined threshold ϵ (i.e., $\left\|\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t] - \mathbf{K}\hat{\mathbf{w}}[t] - \hat{\boldsymbol{\epsilon}}[t]\right\|_2 / \|\mathbf{y}\|_2$). The ADMM iteration reads as follows.

Update the primal variables:

For obtaining $\hat{\mathbf{h}}_i, i \in \{1, \ldots, J\}$, we need to minimise

$$\hat{\mathbf{h}}_i[t+1] = \underset{\hat{\mathbf{h}}_i[t]}{\operatorname{argmin}} \left\| \mathbf{y} - \sum_{k=1}^{J} \mathbf{F}_k \hat{\mathbf{h}}_k[t] - \mathbf{K}\hat{\mathbf{w}}[t] - \hat{\boldsymbol{\epsilon}}[t] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right\|_2^2. \tag{4.21}$$

Problem (4.21) admits a closed-form solution, which is

$$\hat{\mathbf{h}}_i[t+1] = \mathbf{F}_i^T \left(\mathbf{y} - \sum_{k=1, k\neq i}^{J} \mathbf{F}_k \hat{\mathbf{h}}_k[t] - \mathbf{K}\hat{\mathbf{w}}[t] - \hat{\boldsymbol{\epsilon}}[t] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right). \tag{4.22}$$

For obtaining optimal $\hat{\mathbf{w}}$, we need to minimise

$$\hat{\mathbf{w}}[t+1] = \underset{\hat{\mathbf{w}}[t]}{\operatorname{argmin}} \left\| \mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \hat{\boldsymbol{\epsilon}}[t] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right\|_2^2 \tag{4.23}$$

Problem (4.23) admits a closed-form solution of the form

$$\hat{\mathbf{w}}[t+1] = \mathbf{K}^T \left(\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \hat{\boldsymbol{\epsilon}}[t] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right). \tag{4.24}$$

For obtaining optimal $\hat{\boldsymbol{\epsilon}}$, we need to minimise

$$\hat{\boldsymbol{\epsilon}}[t] = \underset{\hat{\boldsymbol{\epsilon}}[t]}{\operatorname{argmin}} \left[\lambda \left\| \mathbf{w}_y \odot \hat{\boldsymbol{\epsilon}}[t] \right\|_1 + \frac{\mu[t]}{2} \left\| \mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right\|_2^2 \right]. \tag{4.25}$$

Utilising the SVT operator, problem (4.25) admits the following solution

$$\hat{\boldsymbol{\epsilon}}[t+1] = \mathbf{w}_y \odot \mathbf{a} + \overline{\mathbf{w}}_y \odot \left(\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right), \tag{4.26}$$

where

$$\mathbf{a} = \mathcal{S}_{\frac{\lambda}{\mu[t]}} \left(\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right). \tag{4.27}$$

Update the Lagrange multiplier and parameter μ:

$$\boldsymbol{\lambda}[t+1] = \boldsymbol{\lambda}[t] + \mu[t] \left(\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] - \hat{\boldsymbol{\epsilon}}[t+1] \right), \tag{4.28}$$

$$\mu[t+1] = \min\left(\rho\mu[t], \mu_{\max}\right). \tag{4.29}$$

Algorithm 7 ADMM solver for problem (4.20)

Input: $\mathbf{y}$, where $\mathbf{y}$ is the test datum, $\mathbf{F}_i, i \in \{1, \ldots, J\}$ and $\mathbf{G}$, where $\mathbf{F}_i$ and $\mathbf{G}$ are the bases extracted utilising Algorithm 6, $\mathbf{K}$ is the linear span of $\mathbf{G}$ given by applying the rank-r SVD on $\mathbf{G}$ and $\mathbf{w}_y$, where $\mathbf{w}_y$ is the occlusion mask corresponding the test datum. **Initialisations:** $t = 0, \epsilon = 1e-8$, maximum number of iterations $t_{\max}, \mu[0] = \frac{25}{\|\mathbf{X}\|}$, $\rho = 1.2$, $\mu_{\max} = 10^7$ and $\lambda = 1/\sqrt{max(F, N)}$. **Output:** the reconstructed UV map without the error term, i.e., $\hat{\mathbf{y}} = \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[T] + \mathbf{K}\hat{\mathbf{w}}[T]$

while *not converged* **do**

  for $i = 1 : J$ **do**

    Update $\hat{\mathbf{h}}_i$:

    $$\hat{\mathbf{h}}_i[t+1] = \mathbf{F}_i^T \left(\mathbf{y} - \sum_{k=1, k\neq i}^{J} \mathbf{F}_k \hat{\mathbf{h}}_k[t] - \mathbf{K}\hat{\mathbf{w}}[t] - \hat{\boldsymbol{\epsilon}}[t] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right)$$

  end

  Update $\hat{\mathbf{w}}$:

  $$\hat{\mathbf{w}}[t+1] = \mathbf{K}^T \left(\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_k \hat{\mathbf{h}}_k[t] - \hat{\boldsymbol{\epsilon}}[t] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right)$$

  Update $\hat{\boldsymbol{\epsilon}}$:

  $$\hat{\boldsymbol{\epsilon}}[t+1] = \mathcal{S}_{\frac{\lambda}{\mu[t]}} \left[\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right]$$

  $$\hat{\boldsymbol{\epsilon}}[t+1] = \mathbf{w}_y \odot \hat{\boldsymbol{\epsilon}}[t+1] + \overline{\mathbf{w}}_y \odot \left[\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] + \frac{\boldsymbol{\lambda}[t]}{\mu[t]} \right]$$

  Update λ:

  $$\boldsymbol{\lambda}[t+1] = \boldsymbol{\lambda}[t] + \mu[t] \left(\mathbf{y} - \sum_{i=1}^{J} \mathbf{F}_i \hat{\mathbf{h}}_i[t+1] - \mathbf{K}\hat{\mathbf{w}}[t+1] - \hat{\boldsymbol{\epsilon}}[t+1] \right)$$

  Update μ:

  $\mu[t+1] = \min(\rho\mu[t], \mu_{\max})$

end

4.3 Experiments

The experimental evaluation of MARCA against other state-of-the-art algorithms is carried out via a series of experiments such as: a) noise analysis on synthetic data, b) noise analysis on real facial UV texture maps, c) age progression "in-the-wild".

In order to extract the incomplete face UV maps that were used in our algorithm we have fitted the various databases with a 3DMM. The 3DMM fitting process that was used is the one

in [116], which is publicly available. The occlusion masks of the fitted images were extracted by utilising the corresponding shape and camera parameters.

For the experimental evaluations, database Multi-PIE [73] and AgeDB [16] (also presented in Chapter 3) were utilised to train MARCA. Multi-PIE is a database captured under controlled lab conditions and thus the images do not contain gross-errors attributed to e.g., occlusions. Nevertheless, Multi-PIE is a multi-attribute database, since it contains labels for attributes such as *identity*, *expression* and *illumination*. That renders it suitable to be utilised in MARCA. In particular, in the training phase of MARCA on Multi-PIE, 90% of the total number of UVs pertaining to distinct identities, expressions and illuminations were utilised. The rest were used for testing.

AgeDB contains images captured under "in-the-wild" conditions (i.e., occlusions, various poses, pixel corruptions are present in the images). Moreover, it is annotated for multiple attributes (i.e., *identity*, *age*) and thus it is suitable for evaluating MARCA. AgeDB was split in six distinct age-groups, namely *21-30*, *31-40*, *41-50*, *51-60*, *61-70* and *71-100*. Then, the UVs belonging to each age-group were further split according to the attribute *identity*. In the training phase of MARCA, 90% of the total number of UVs were kept to extract the bases with respect to attributes *identity* and *age-groups* and the rest were used for testing.

4.3.1 Synthetic noise analysis

In this section we detail the experiments we conducted on synthetic data contaminated with artificial noise of different forms and levels. We report quantitative results and compare the performance of MARCA against RJIVE [122], RPCA-L1 [46] and OM-RPCA [120].

In more detail, with the following experiments we aim to investigate the ability of the four component analysis methods to handle data that have been contaminated with two common types of noise. Sparse noise, which is composed of errors that are sparsely supported but of

large or unbounded magnitude, is a form of noise that is often present in visual data. Such noise can be the salt & pepper noise, occlusions and registration errors. However, errors of small magnitude, like ambient noise or quantisation noise cannot be assumed as sparse errors. For such types of noise it is reasonable to assume that they follow a Gaussian distribution of small variance. Thus, the two types of noise that we consider in our experiments are the gross, sparse, non-Gaussian noise and the Gaussian noise. We conduct experiments for different levels of each form of noise mentioned above.

We generate the data as follows. We create a random matrix $\mathbf{X} \in \mathbf{R}^{m \times n}$ for various dimensions that admits the following decomposition: $\mathbf{X} = \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* + \mathbf{E}_*$, where $\mathbf{F}_*\mathbf{H}_*$ and $\mathbf{G}_*$ are random matrices with ranks r_1 and r_2, respectively. $\mathbf{E}_*$ corresponds to the noise term. As mentioned previously, we used different noise forms (i.e., sparse non-Gaussian and Gaussian) as well as different noise levels per form. More specifically, we present the results when applying MARCA, RJIVE [122], RPCA-L1 [46] and OM-RPCA [120] to extract the error-free components for the following cases:

- *Sparse, non-Gaussian* noise, contaminating $10\%, 25\%$ and 50% of the data matrix entries. The values for the regulariser λ used for the experiments are provided by the formula: $\lambda = 1/\sqrt{max(m,n)}$, following the selection process described in [33]. In particular, for the case of $(m,n) = (500, 500)$, we used $\lambda \approx 0.04$ and for the case of $(m,n) = (1000, 1000)$, we used $\lambda \approx 0.03$. The convergence threshold ϵ used for this experiment for MARCA and RJIVE [122] is $1e-8$. In Table 5.1 we report the reconstruction losses where, as evinced, MARCA successfully reconstructs the error-free components. In the contrary, RPCA-L1 [46] and OM-RPCA [120] perform poorly compared to RJIVE [122] and MARCA, due to the existence of the term $\mathbf{F}_*\mathbf{H}_*$, which is a matrix with identical columns in many parts (e.g., if $\mathbf{F}_*\mathbf{H}_*$ referred to the age-group component, the columns corresponding to the age-group *21-30* would be the same for all of the samples that belong in the specific age-group). The specific structure of $\mathbf{F}_*\mathbf{H}_*$ does not comply with the incoherence property

of the low-rank component in the RPCA [33] setting, since $\mathbf{F}_*\mathbf{H}_*$ is a matrix with sparse singular values. As a result, since both RPCA-L1 [46] and OM-RPCA [120] belong in the family of RPCA algorithms, the poor performance under this setting is expected. We empirically validated this fact by conducting synthetic experiments under the setting where the incoherence property for the low-rank component holds.

- *Gaussian* noise corresponding to zero-mean samples with variances of $0.1, 0.25$ and 0.5. The value for the regulariser λ used for the experiments is very large, i.e., $\lambda = 10,000$, since MARCA can handle data contaminated with Gaussian noise by vanishing the sparse error term, as explained in Section 4.2.3. The convergence threshold ϵ used for this experiment for MARCA and RJIVE [122] is $1e-8$. In Table 5.3 we report the reconstruction losses where, as evinced, all the four methods successfully reconstruct the error-free components.

We should note that since MARCA constitutes the first multi-attribute learning algorithm, we ran experiments under the single attribute scenario (i.e., only a single term $\mathbf{F}_*\mathbf{H}_*$ is utilised), so that comparisons against the other methods are feasible.

4.3.2 Qualitative noise analysis

In this section we detail the experiments we conducted on real data contaminated with artificial noise of different forms and levels. We report qualitative results and compare the performance of MARCA against RJIVE [122], RPCA-L1 [46] and OM-RPCA [120].

To pursue this analysis we employ the UV texture data from images of neutral expression from Multi-PIE [73]. Five images of the same person, captured simultaneously in five different viewing angles, $-30°$,$-15°$, $0°$, $15°$ and $30°$ were used to obtain the UV textures. We first fitted a 3DMM on the five images of each person and then generated a corresponding UV map from each image. By using the occlusion-free pixels of each of the five UV maps we

Table 4.1: Quantitative results when comparing reconstruction losses of the error-free components as retrieved by MARCA against RPCA-L1 [46], OM-RPCA [120] and RJIVE [122] on synthetic data of various dimensions, contaminated with sparse, non-Gaussian noise, where the data decomposition is $\mathbf{X} = \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* + \mathbf{E}_*$. In the case of RJIVE [122], $\mathbf{FH}$ corresponds to the individual component $\mathbf{A}$ and $\mathbf{G}$ to the joint component $\mathbf{J}$. In the case of RPCA-L1 [46] and OM-RPCA [120], the projection matrix $\mathbf{W}$ is related to the low rank component $\mathbf{FH}+\mathbf{G}$. All of the experiments were executed on an Intel i9-7900X 3.30GHz machine. We ran each experiment 10 times and report the average execution time for each one.

(m, n, r_1, r_2)	Method	$\frac{\lVert \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* - \mathbf{FH} - \mathbf{G} \rVert_F^2}{\lVert \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* \rVert_F^2}$			Execution time (s)		
	Noise level	10%	25%	50%	10%	25%	50%
(500, 500, 5, 10)	RPCA-L1 [46]	0.182	0.313	2.159	0.99	1.09	0.84
	OM-RPCA [120]	0.181	0.313	2.159	0.36	0.36	0.37
	RJIVE [122]	$3.36e-09$	$3.62e-09$	$6.17e-04$	25.75	26.33	24.38
	MARCA	$1.78e-10$	$4.81e-10$	$2.56e-04$	16.25	17.27	16.83
(1000, 1000, 10, 20)	RPCA-L1 [46]	0.183	0.315	2.167	5.04	4.17	4.64
	OM-RPCA [120]	0.183	0.315	2.167	1.72	1.80	1.84
	RJIVE [122]	$1.07e-09$	$2.27e-09$	$8.25e-04$	71.18	71.98	65.11
	MARCA	$3.56e-10$	$5.75e-10$	$4.39e-04$	24.85	27.28	28.30

$-30°$ $-15°$ $0°$ $15°$ $30°$ full UV

Figure 4.4: UV texture maps extracted by five images of the same person, captured simultaneously from different viewing angles and contained in the multipie database. By combining the five UV maps we are able to generate a full UV texture map.

were able to obtain a full UV texture map that contains no missing values as can also be seen in Fig. 4.4. We trained all of the four algorithms on UV maps that are completely free of sparse errors (i.e., we only chose subjects that are *not* wearing eye-glasses and also created UV maps without missing values). This is of important essence, as we can then artificially add noise on the error-free UV maps which in turn can be used as ground truth for evaluating the performance of the four methods.

Table 4.2: Quantitative results when comparing reconstruction losses of the error-free components as retrieved by MARCA against RPCA-L1 [46], OM-RPCA [120] and RJIVE [122] on synthetic data of various dimensions, contaminated with Gaussian noise, where the data decomposition is $\mathbf{X} = \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* + \mathbf{E}_*$. In the case of RJIVE [122], $\mathbf{FH}$ corresponds to the individual component $\mathbf{A}$ and $\mathbf{G}$ to the joint component $\mathbf{J}$. In the case of RPCA-L1 [46] and OM-RPCA [120], the projection matrix $\mathbf{W}$ is related to the low rank component $\mathbf{FH} + \mathbf{G}$. All of the experiments were executed on an Intel i9-7900X 3.30GHz machine. We ran each experiment 10 times and report the average execution time for each one.

(m, n, r_1, r_2)	Method	$\frac{\lVert \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* - \mathbf{FH} - \mathbf{G} \rVert_F^2}{\lVert \mathbf{F}_*\mathbf{H}_* + \mathbf{G}_* \rVert_F^2}$			Execution time (s)		
	Variance level	0.10	0.25	0.50	0.10	0.25	0.50
(500, 500, 5, 10)	RPCA-L1 [46]	$6.89e-09$	$4.32e-08$	$1.71e-07$	0.36	0.35	0.35
	OM-RPCA [120]	$6.89e-09$	$4.32e-08$	$1.71e-07$	0.38	0.38	0.36
	RJIVE [122]	$6.89e-09$	$4.32e-08$	$1.71e-07$	31.95	32.34	28.53
	MARCA	$6.89e-09$	$4.32e-08$	$1.71e-07$	18.34	18.00	17.65
(1000, 1000, 10, 20)	RPCA-L1 [46]	$1.88e-09$	$1.17e-08$	$4.70e-07$	2.39	1.81	1.83
	OM-RPCA [120]	$1.88e-09$	$1.17e-08$	$4.70e-07$	1.84	1.82	1.84
	RJIVE [122]	$1.88e-09$	$1.17e-08$	$4.70e-07$	142.78	142.03	140.39
	MARCA	$1.88e-09$	$1.17e-08$	$4.70e-07$	91.83	86.98	85.04

We then applied MARCA, RJIVE [122], RPCA-L1 [46] and OM-RPCA [120] to extract the error-free components. Similarly to the quantitative experiments, we present the results for the following cases:

- *Sparse, non-Gaussian* noise with error terms corresponding to samples with $10\%, 25\%$ and 50% non-zero entries. The value for the regulariser λ used for the experiments is provided by the formula: $\lambda = 1/\sqrt{max(m, n)}$, following the selection process described in [33]. In particular, for $n = 765$ UV images and $m = 556266$ the size of a vectorised UV image, $\lambda \approx 0.0015$. The convergence threshold ϵ used for this experiment for MARCA and RJIVE [122] is $1e-5$, which is sufficient for the analysis of images. The recovered rank of the individual subspace $\mathbf{G}$ when MARCA was applied was 185 for 10% noise, 197 for 25% noise and 216 for 50% noise. These rank values are expected as $\mathbf{G}$ is by

definition a low rank matrix which is able to express the individual variations of the data. Also, the low rank of matrix $\mathbf{G}$ is sensible for the data in use as all data are facial images with similar background. As can been seen in Fig. 4.5, MARCA is able to produce better results in recovering the error free images than RPCA-L1 [46] and OM-RPCA [120] in almost all cases. In comparison to RJIVE [122], MARCA retains the identity and illumination with greater success.

- *Gaussian* noise with error terms corresponding to zero-mean samples with variances of $0.1, 0.25$ and 0.5. The value for the regulariser λ used for the experiments is very large, i.e., $\lambda = 10,000$, since MARCA can handle data contaminated with Gaussian noise by vanishing the sparse error term, as explained previously. The convergence threshold ϵ used for this experiment for MARCA and RJIVE [122] is $1e-5$, which is sufficient for the analysis of images. The recovered rank of the individual subspace $\mathbf{G}$ when MARCA was applied was 164 for variance 0.1, 182 for variance 0.25 and 214 for variance 0.5. These rank values are expected as in the previous experiment. As anticipated, all four methods are able to recover the error-free images with high accuracy as is also shown in Fig. 4.6. RPCA-L1 [46] seems to have produced the lowest quality reconstructions.

Similarly to Section 4.3.1, we should note that since MARCA constitutes the first multi-attribute learning algorithm, we ran experiments under the single attribute scenario (i.e., only a single term $\mathbf{F}_*\mathbf{H}_*$ is utilised), so that comparisons against the other methods are feasible.

4.3.3 Age-progression "in-the-wild"

Age-progression "in-the-wild" entails the task of rendering a facial image of a subject at various ages. It is arguably a very challenging task in Computer Vision, since "in-the-wild" images are captured in uncontrolled conditions (e.g., different illuminations and poses, self-occlusions, etc.). AgeDB [16], also presented in Chapter 3, was utilised in this experiment, since it is a

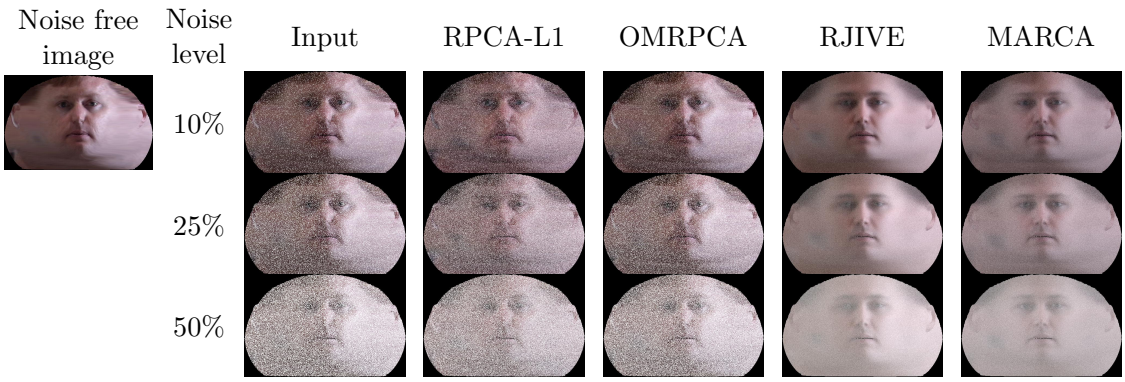


Figure 4.5: Reconstructed UV texture maps produced by employing RPCA-L1, OM-RPCA, RJIVE and MARCA on data that have been contaminated with different levels of sparse, non-Gaussian noise.

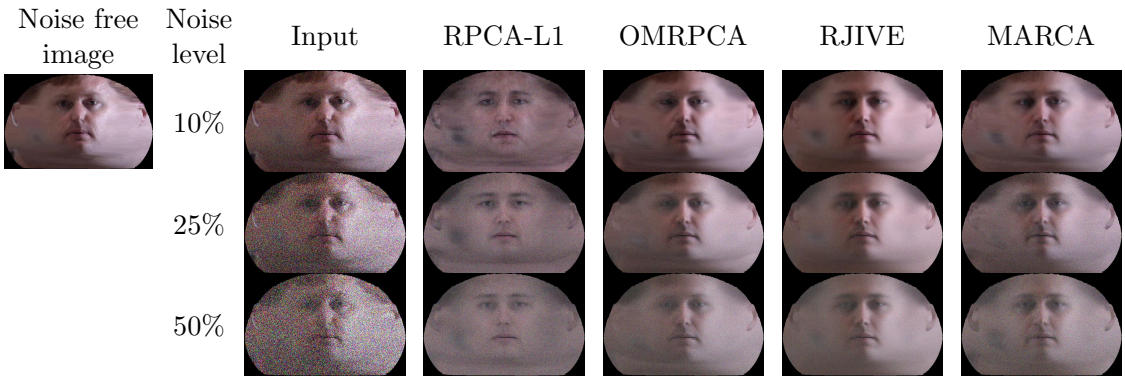


Figure 4.6: Reconstructed UV texture maps produced by employing RPCA-L1, OM-RPCA, RJIVE and MARCA on data that have been contaminated with different levels of Gaussian noise.

manually collected "in-the-wild" age database with accurate *age* and *identity* labels and hence the extracted age-groups and identity bases will contain no errors due to incorrect annotations.

During the training phase, MARCA was trained under the multi-attribute scenario, incorporating both the knowledge of the *age* as well as the *identity* attributes. In the testing phase, a random UV map which did not belong to the training set was chosen and reconstructed for various ages following the process described in Section 4.2.4.

Comparisons against other broadly used age-progression methods are provided in Fig. 4.7 and Fig. 4.8. More specifically, we compare MARCA against Illumination Aware Age Pro-

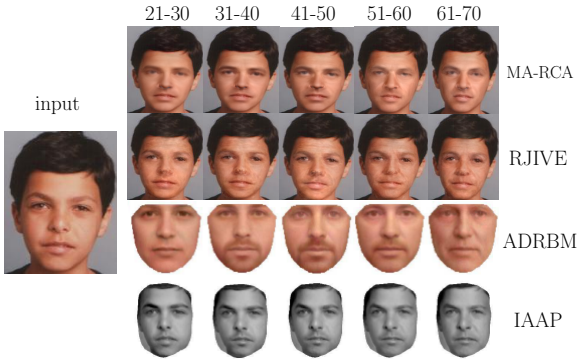


Figure 4.7: Age-progression results on FG-Net database [91] produced by IAAP, ADRBM, RJIVE and the proposed MARCA method. RJIVE and MARCA are trained on AgeDB [16].

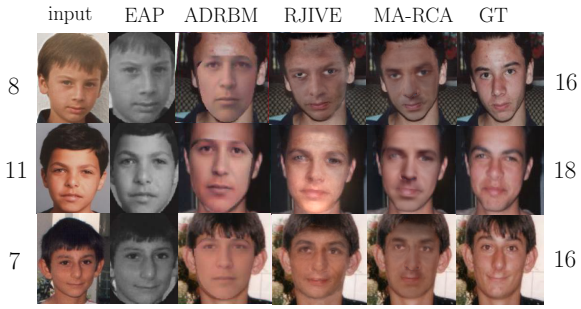


Figure 4.8: Age-progression results produced by EAP, ADRBM, RJIVE and the proposed MARCA method on images of the FG-Net database [91]. MARCA and RJIVE are trained on AgeDB [16].

gression (IAAP) [106], Aging with Deep Restricted Boltzmann Machines (ADRBM) [108], Exemplar-based Age Progression (EAP) [127] and RJIVE [122] . Finally, MARCA is compared against the state-of-the-art RJIVE [122] in Fig. 4.9.

4.4 Conclusions

With the use of 3D face fitting methods we can generate incomplete facial UV maps of the facial textures. The use of incomplete facial UV maps contaminated with gross errors introduces many challenges and opportunities. In particular, since facial UV maps lie in a pose-free space, linear component analysis techniques can be applied to learn statistical components for various

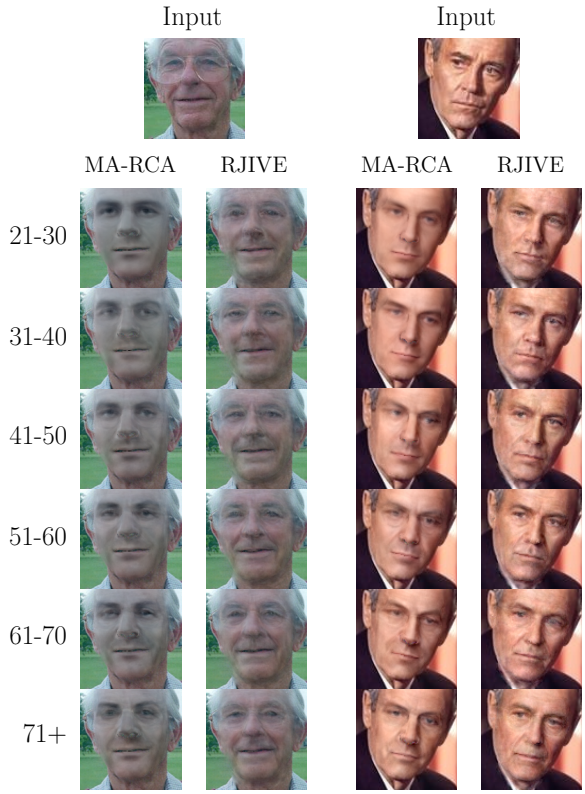


Figure 4.9: Comparing MARCA against state-of-the-art RJIVE in age-progression "in-the-wild" experiments. As can be seen in both cases, MARCA reconstructions are more realistic compared to the reconstructions utilising RJIVE.

tasks. In this Chapter, we presented a novel statistical robust component analysis technique that can tackle the above challenges and at the same time exploit multiple labels of the data at hand during training.

Chapter 5

Multi-Attribute Probabilistic Linear Discriminant Analysis for 3D Facial Shapes

Contents

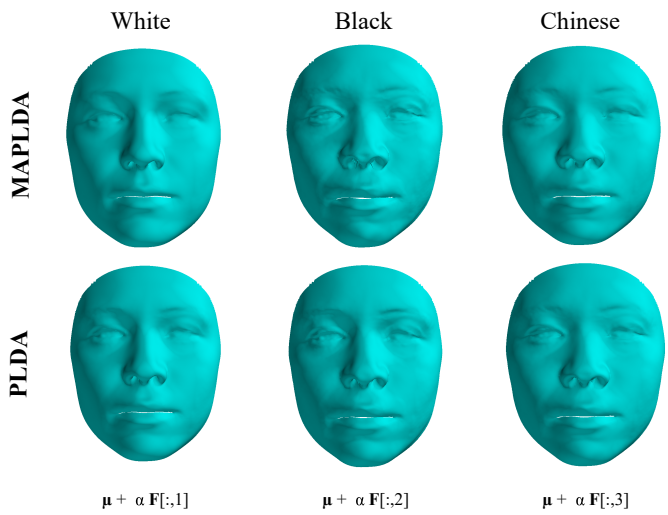


Figure 5.1: Visualisation of recovered components by MAPLDA as compared to PLDA, highlighting the improvement induced by explicitly accounting for multiple attributes. We denote with $\boldsymbol{\mu}$ the global mean, and with $\mathbf{F}$ the learned subspace of the *ethnicity* attribute where $\alpha \geq 1$ is used to accentuate the component visualisation. MAPLDA is trained by jointly taking into account the *ethnicity* and *age-group* attributes. As can be seen, this leads to a more accurate representation of the *ethnicity attribute* in MAPLDA, which is more prominent for the *Black* class.

5.1 Introduction

CA techniques such as PCA [128], LDA [129] and Canonical Correlation Analysis (CCA) [130] are among the most popular methods for feature extraction and dimensionality reduction, typically utilised in a wide range of applications in Computer Vision and Machine Learning. While CA methods such as PCA have been introduced in the literature more than a century ago, it was only during the last two decades that *probabilistic* interpretations of CA techniques have been introduced in the literature, with examples of such efforts including Probabilistic PCA (PPCA) [131, 132, 133], Probabilistic LDA (PLDA) [134, 135, 136, 137, 8, 9] and Probabilistic CCA (PCCA) [138, 139]. The rise in popularity of probabilistic CA methods can be

attributed to several appealing properties, such as explicit variance modelling and inherent handling of missing data [140]. Furthermore, probabilistic CA models may be easily extended to mixture models [141] and Bayesian methodologies [142], while they can also be utilised as general density models [133].

While many CA methods such as PCA and CCA are typically considered to be unsupervised, methods such as LDA assume knowledge of labelled data in order to derive a discriminative subspace based on attribute values (labels), that can subsequently be utilised for predictive analysis e.g., classification of unlabelled data. Probabilistic LDA (PLDA) [9, 143] constitutes one of the first attempts towards formulating a probabilistic generative CA model that incorporates information regarding data labels (e.g., the identity of a person in an image). In more detail, as have also mentioned in the introductory Chapter, each datum is generated by two distinct subspaces: a subspace that incorporates information among instances belonging to the same class, and a subspace that models information that is unique to each datum. Put simply in the context of face recognition, all images of a specific subject share the same identity, while each image may carry its own particular variations (e.g., in terms of illumination, pose and so on).

Nevertheless, a feature of PLDA and other probabilistic LDA variants that can be disadvantageous is the *single-attribute* assumption. In other words, PLDA is limited to the knowledge of one attribute, effectively disregarding knowledge of any other attributes available for the data at hand that may prove beneficial for a given task. For example, it is reasonable to assume that knowledge of attributes such as *pose*, *expression* and *age* may be deemed beneficial in terms of determining the identity of a person in a facial image. By incorporating knowledge of multiple attributes, we would expect a generative model to better explain the observation variance, by decomposing the observation space into multiple components conditioned on the attributes at hand. Fig. 5.1 illustrates the more accurate representations we can obtain in this way.

In the past, PLDA was successfully applied to tasks such as face recognition and speaker verification [9, 144]. The advent of Deep Convolutional Neural Networks (DCNNs) provided models that overperformed linear CA techniques with respect to feature extraction in Computer Vision applications that involve intensity images and video, mainly due to the complex variations introduced by the texture and the geometric transformations. Nevertheless, linear CA techniques remain prominent and powerful techniques for tasks related to the analysis of 3D shapes, especially in case that dense correspondences have been established among them. Recently, very powerful frameworks have been proposed for establishing dense correspondences in large scale databases of 3D faces [145, 146], 3D bodies [147] and 3D hands [148].

Given that several modern databases of 3D shapes are annotated in terms of multiple attributes, and further motivated by the aforementioned shortcomings of single-attribute methods, in this Chapter we present a Multi-Attribute generative probabilistic variant of LDA, dubbed Multi-Attribute Probabilistic LDA (MAPLDA). The proposed MAPLDA is able to *jointly* model the influence of multiple attributes on observed data, thus effectively decomposing the observation space into a set of subspaces depending on multiple attribute instantiations. As shown via a set of experiments on age, ethnicity and age group identification, the joint multi-attribute modelling embedded in MAPLDA appears highly beneficial, outperforming other single-attribute approaches in an elegant probabilistic framework. In what follows, we briefly summarise the contributions of our work.

- We present MAPLDA, the first probabilistic variant of LDA that is *inherently* able to *jointly* model multiple attributes.
- We provide a probabilistic formulation and optimisation procedure for training, as well as a flexible framework for performing inference on *any* subset of the multiple attributes available during training.
- We demonstrate the advantages of joint-attribute modelling by a set of experiments on

the MeIn3D dataset [146], in terms of ethnicity, age and weight group identification, as well as facial shape generation.

The rest of the Chapter is organised as follows. In Section 5.2, we briefly recap PLDA, a generative counterpart to LDA. MAPLDA is introduced in Section 5.3, along with details on optimisation and inference. Finally, experimental evaluation is detailed in Section 5.4.

5.2 Probabilistic Linear Discriminant Analysis

In this section, we briefly review the PLDA model introduced in [9, 143] and presented in more detail in Chapter 1. As aforementioned, PLDA carries the assumption that data are generated by two different subspaces: one that depends on the class and one that depends on the sample. That is, assuming we have a total of I classes, with each class i containing a total of J samples, then the j-th datum of the i-th class is defined as:

$$\mathbf{x}_{i,j} = \boldsymbol{\mu} + \mathbf{F}\mathbf{h}_i + \mathbf{G}\mathbf{w}_{i,j} + \boldsymbol{\epsilon}_{i,j} \tag{5.1}$$

where $\boldsymbol{\mu}$ denotes the global mean of the training set, $\boldsymbol{F}$ defines the subspace capturing the identity of every subject, with $\boldsymbol{h}_i$ being the latent identity variable representing the position in the particular subspace. Furthermore, $\boldsymbol{G}$ defines the subspace modelling variations among data, with $\boldsymbol{w}_{i,j}$ being the associated latent variable. Finally, $\boldsymbol{\epsilon}_{i,j}$ is a residual noise term which is Gaussian with diagonal covariance $\boldsymbol{\Sigma}$. Assuming zero-mean observations, the model in (5.1) can be described as:

$$P\left(\boldsymbol{x}_{i,j}|\boldsymbol{h}_i, \boldsymbol{w}_{i,j}, \boldsymbol{\theta}\right) = \mathcal{N}_{\boldsymbol{x}}\left(\boldsymbol{F}\boldsymbol{h}_i + \boldsymbol{G}\boldsymbol{w}_{i,j}, \boldsymbol{\Sigma}\right) \tag{5.2}$$

$$P\left(\boldsymbol{h}_i\right) = \mathcal{N}_{\boldsymbol{h}}\left(\mathbf{0}, \boldsymbol{I}\right) \tag{5.3}$$

$$P\left(\boldsymbol{w}_{i,j}\right) = \mathcal{N}_{\boldsymbol{w}}\left(\mathbf{0}, \boldsymbol{I}\right) \tag{5.4}$$

where the set of parameters $\boldsymbol{\theta} = \{\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Sigma}\}$ is optimised during training via EM [49]. In the training process, EM is applied and the optimal set of parameters, $\boldsymbol{\theta} = \{\boldsymbol{F}, \boldsymbol{G}, \boldsymbol{\Sigma}\}$, is recovered.

5.3 Multi-Attribute PLDA (MAPLDA)

Let us consider a generalisation of the single-attribute setting, as described in Section 5.2. In particular, let us assume that the data at hand are labelled in terms of a total of N attributes, where each attribute may take K_i discrete instantiations (labels/classes), that is $a_i \in \{1, \cdots, K_i\}$[1]. We further assume that a set of J data available during training for any distinct combination of attribute instantiations. The generative model for MAPLDA corresponding to the j-th observation (datum) can then be described as:

$$\boldsymbol{x}_{a_{1:N},j} = \boldsymbol{\mu} + \sum_{i=1}^{N} \boldsymbol{F}_i \boldsymbol{h}_{i,a_i} + \boldsymbol{G}\boldsymbol{w}_{a_{1:N},j} + \boldsymbol{\epsilon}_{a_{1:N},j} \tag{5.5}$$

where $\boldsymbol{\mu}$ denotes the training set global mean, $\boldsymbol{F}_1, \ldots, \boldsymbol{F}_N$ are loadings that define the subspace bases for each particular attribute (e.g., $\boldsymbol{F}_1$ may be the basis for the attribute age-group, $\boldsymbol{F}_2$ the basis for the attribute ethnicity, etc.) and $\boldsymbol{h}_{1,a_1}, \ldots, \boldsymbol{h}_{N,a_N}$ are selectors that define the position in each subspace, respectively (e.g., selector $\boldsymbol{h}_{1,a_1}$ will render the distinct age-group instantiation with which each datum is annotated). Furthermore, the matrix $\boldsymbol{G}$ defines a basis for the subspace that models the variations among the data and $\boldsymbol{w}_{a_{1:N},j}$ defines the position in that subspace for the j-th datum. Finally, random noise is captured through the term $\boldsymbol{\epsilon}_{a_1:N,j}$ which is specific for each datum and is set as a Gaussian with diagonal covariance $\boldsymbol{\Sigma}$. Note that from here on, to avoid cluttering the notation we omit dependence on attribute instantiations (unless specified otherwise), that is we denote $\boldsymbol{x}_{a_{1:N},j}$ as $\boldsymbol{x}_j$, $\boldsymbol{w}_{a_{1:N},j}$ as $\boldsymbol{w}_j$ and $\boldsymbol{\epsilon}_{a_{1:N},j}$ as $\boldsymbol{\epsilon}_j$. Moreover, by assuming zero-mean observations, the model in (5.5) can be written more clearly as:

$$\boldsymbol{x}_j = \sum_{i=1}^{N} \boldsymbol{F}_i \boldsymbol{h}_{i,a_i} + \boldsymbol{G}\boldsymbol{w}_j + \boldsymbol{\epsilon}_j \tag{5.6}$$

while the prior probabilities of (5.6) can be written as:

$$P\left(\boldsymbol{h}_{i,a_i}\right) = \mathcal{N}_{\boldsymbol{h}}\left(\boldsymbol{0}, \boldsymbol{I}\right), \quad \forall i \in \{1, \ldots, N\} \tag{5.7}$$

$$P\left(\boldsymbol{w}_j\right) = \mathcal{N}_{\boldsymbol{w}}\left(\boldsymbol{0}, \boldsymbol{I}\right) \tag{5.8}$$

[1]For brevity of notation, we denote $a_1, \ldots, a_N$ as $a_{1:N}$.

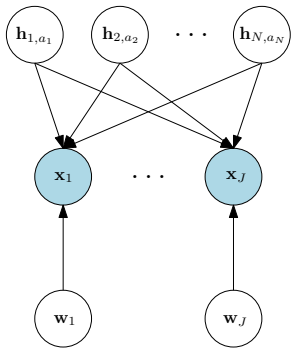


Figure 5.2: Graphical model for J observed data of the training set (i.e., $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_J$) for a distinct combination of attribute instantiations. The positions of the data in the subspaces $\boldsymbol{F}_1, \ldots, \boldsymbol{F}_N$ are given by the latent variables $\boldsymbol{h}_{1,a_1}, \ldots, \boldsymbol{h}_{N,a_N}$, respectively, while the position in subspace $\boldsymbol{G}$ is given by the latent variables $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_J$, respectively.

and the posterior as:

$$P\left(\boldsymbol{x}_j | \boldsymbol{h}_{1,a_1}, \ldots, \boldsymbol{h}_{N,a_N}, \boldsymbol{w}_j, \boldsymbol{\theta}\right) = \mathcal{N}_{\boldsymbol{x}}\left(\sum_{i=1}^{N} \boldsymbol{F}_i \boldsymbol{h}_{i,a_i} + \boldsymbol{G}\boldsymbol{w}_j, \boldsymbol{\Sigma}\right) \tag{5.9}$$

where $\boldsymbol{\theta} = \{\boldsymbol{F}_1, \ldots, \boldsymbol{F}_N, \boldsymbol{G}, \boldsymbol{\Sigma}\}$ is the set of parameters. Having defined our model, in the next subsections we detail both the training and inference procedures of MAPLDA in the presence of multiple attributes. For further clarification, we note that the graphical model of MAPLDA is illustrated in Fig. 5.2.

5.3.1 Training with Multiple Attributes

In this section, we detail the estimation of both the latent variables and parameters involved in MAPLDA. We assume that we are interested in making predictions regarding a subset of available attributes. While any subset can be chosen, for purposes of clarity and without loss of generality, we assume this set consists of the first $N-1$ attributes. That is, when given a test datum we can assign any of the $N-1$ attributes to classes $a_i, i \in \{1, \ldots, K_i\}$, while exploiting the knowledge of the remaining attributes (e.g., by marginalisation during inference). Furthermore, without loss of generality, assume that there is a total of M data for each distinct combination of the $N-1$ attributes instantiations. We denote $\mathbf{F} \doteq \begin{bmatrix} \mathbf{F}_1 & \mathbf{F}_2 & \ldots & \mathbf{F}_{N-1} \end{bmatrix}$, and

$\mathbf{h} \doteq \begin{bmatrix} \mathbf{h}_{1,a_1}^T & \mathbf{h}_{2,a_2}^T & \dots & \mathbf{h}_{N-1,a_{N-1}}^T \end{bmatrix}^T$ the block matrices consisting of loadings and variables for the first N-1 attributes, and $\hat{\mathbf{h}}_N \doteq \begin{bmatrix} \mathbf{h}_{N,1}^T & \mathbf{h}_{N,2}^T & \dots & \mathbf{h}_{N,K_N}^T \end{bmatrix}^T$ the latent variable block matrix for all attribute values of the N-th attribute. Following a block matrix formulation, we group the M data samples as follows,

$$\begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_M \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} & e_{1,a_N} \otimes \mathbf{F}_N & \boldsymbol{G} & \mathbf{0} & \dots & \mathbf{0} \\ \boldsymbol{F} & e_{2,a_N} \otimes \mathbf{F}_N & \mathbf{0} & \boldsymbol{G} & \dots & \mathbf{0} \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \boldsymbol{F} & e_{M,a_N} \otimes \mathbf{F}_N & \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \boldsymbol{h} \\ \hat{\mathbf{h}}_N \\ \boldsymbol{w}_1 \\ \vdots \\ \boldsymbol{w}_M \end{bmatrix} + \begin{bmatrix} \boldsymbol{\epsilon}_1 \\ \vdots \\ \boldsymbol{\epsilon}_M \end{bmatrix} \tag{5.10}$$

where $\otimes$ denotes the Kronecker product, and $e_{i,a_N} \in \mathbb{R}^{1\times K_N}$ is a one-hot embedding of the value of attribute a_N for datum $\mathbf{x}_i$ (recall that $a_N \in \{1, \dots, K_N\}$). For example, assume that for $\mathbf{x}_1$, $a_N = K_N$. Then, $e_{1,N} = [0, \dots, 0, 1] \in \mathbb{R}^{1\times K_N}$ and $e_{1,N} \otimes \mathbf{F}_N = [\mathbf{0}, \dots, \mathbf{0}, \mathbf{F}_N]$. Furthermore, (5.10) can be written compactly as:

$$\boldsymbol{x}' = \boldsymbol{A}\boldsymbol{y} + \boldsymbol{\epsilon}' \tag{5.11}$$

where the prior and conditional probabilities of (5.11) can now be written as:

$$P\left(\boldsymbol{x}'|\boldsymbol{y}, \boldsymbol{\theta}\right) = \mathcal{N}_{\boldsymbol{x}'}\left(\boldsymbol{A}\boldsymbol{y}, \boldsymbol{\Sigma}'\right) \tag{5.12}$$

$$P\left(\boldsymbol{y}\right) = \mathcal{N}_{\boldsymbol{y}}\left(\mathbf{0}, \boldsymbol{I}\right) \tag{5.13}$$

where:

$$\boldsymbol{\Sigma}' = \begin{bmatrix} \boldsymbol{\Sigma} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \boldsymbol{\Sigma} \end{bmatrix} \tag{5.14}$$

Following EM and given an instantiation of the model parameters $\boldsymbol{\theta} = \{\boldsymbol{F}_1, \dots, \boldsymbol{F}_N, \boldsymbol{G}, \boldsymbol{\Sigma}\}$, we need to estimate the sufficient statistics, that is the first and second moments of the posterior latent distribution $P(\boldsymbol{y}|\boldsymbol{x}', \boldsymbol{\theta})$. Since both (5.12) and (5.13) refer to Gaussian distributions, it can easily be shown [149] that the posterior also follows a Gaussian distribution:

$$P(\boldsymbol{y}|\boldsymbol{x}', \boldsymbol{\theta}) = \mathcal{N}_{\boldsymbol{y}}\left(\hat{\mathbf{A}}\boldsymbol{A}^T\boldsymbol{\Sigma}'^{-1}\boldsymbol{x}', \hat{\mathbf{A}}\right) \tag{5.15}$$

where $\hat{\mathbf{A}} \doteq \left(\boldsymbol{A}^T\boldsymbol{\Sigma}'^{-1}\boldsymbol{A} + \boldsymbol{I}\right)^{-1}$, and thus:

$$\mathbb{E}[\boldsymbol{y}] = \hat{\mathbf{A}}\boldsymbol{A}^T\boldsymbol{\Sigma}'^{-1}\boldsymbol{x}' \tag{5.16}$$

$$\mathbb{E}\left[\boldsymbol{y}\boldsymbol{y}^T\right] = \hat{\mathbf{A}} + \mathbb{E}[\boldsymbol{y}]\,\mathbb{E}[\boldsymbol{y}]^T \tag{5.17}$$

Having derived the sufficient statistics of MAPLDA, we carry on to the maximisation step. In order to recover the parameter updates, we take the partial derivatives of the conditional (on the posterior) expectation of the complete-data log likelihood of MAPLDA with regards

to parameters $\boldsymbol{\theta} = \{\boldsymbol{F}_1, \ldots, \boldsymbol{F}_N, \boldsymbol{G}, \boldsymbol{\Sigma}\}$. In order to do so, we firstly rewrite (5.6) as follows:

$$\boldsymbol{x}_j = \begin{bmatrix} \boldsymbol{F}_1 & \ldots & \boldsymbol{F}_N & \boldsymbol{G} \end{bmatrix} \begin{bmatrix} \boldsymbol{h}_{1,a_1} \\ \vdots \\ \boldsymbol{h}_{N,a_N} \\ \boldsymbol{w}_j \end{bmatrix} + \boldsymbol{\epsilon}_j \tag{5.18}$$

where (5.18) can be compactly written as:

$$\boldsymbol{x}_j = \boldsymbol{B}\boldsymbol{z}_j + \boldsymbol{\epsilon}_j. \tag{5.19}$$

By adopting the aforementioned grouping, our set of parameters is now denoted as $\boldsymbol{\theta} = \{\boldsymbol{B}, \boldsymbol{\Sigma}\}$, and the complete-data log likelihood conditioned on the posterior is formulated as:

$$\mathcal{Q}\left(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}\right) = \sum_{\boldsymbol{Z}} P\left(\boldsymbol{Z}|\boldsymbol{X}, \boldsymbol{\theta}^{old}\right) \ln\left[P\left(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}\right)\right] \tag{5.20}$$

where the joint can be decomposed as:

$$P\left(\boldsymbol{X}, \boldsymbol{Z}|\boldsymbol{\theta}\right) = \prod_{a_1=1}^{K_1} \cdots \prod_{a_N=1}^{K_N} \prod_{j=1}^{J} P\left(\boldsymbol{x}_{a_{1:N},j}|\boldsymbol{z}_{a_{1:N},j}\right) P\left(\boldsymbol{z}_{a_{1:N},j}\right) \tag{5.21}$$

It can be easily shown [149] that the updates are as follows:

$$\boldsymbol{B} = \left(\sum_{a_1=1}^{K_1} \cdots \sum_{a_N=1}^{K_N} \sum_{j=1}^{J} \boldsymbol{x}_{a_{1:N},j} \mathbb{E}\left[\boldsymbol{z}_{a_{1:N}}\right]^T\right) \left(\sum_{a_1=1}^{K_1} \cdots \sum_{a_N=1}^{K_N} \mathbb{E}\left[\boldsymbol{z}_{a_{1:N}} \boldsymbol{z}_{a_{1:N}}^T\right]\right)^{-1} \tag{5.22}$$

$$\boldsymbol{\Sigma} = \frac{1}{\mathcal{K}J} \mathrm{Diag}\left(\boldsymbol{S}_t - \boldsymbol{B} \sum_{a_1=1}^{K_1} \cdots \sum_{a_N=1}^{K_N} \sum_{j=1}^{J} \mathbb{E}\left[\boldsymbol{z}_{a_{1:N}}\right] \boldsymbol{x}_{a_{1:N},j}^T\right), \tag{5.23}$$

with $\boldsymbol{S}_t = \sum_{a_1=1}^{K_1} \cdots \sum_{a_N=1}^{K_N} \sum_{j=1}^{J} \boldsymbol{x}_{a_{1:N},j} \boldsymbol{x}_{a_{1:N},j}^T$ being the total covariance matrix and $\mathcal{K} = \prod_{i=1}^{N} K_i$.

5.3.2 Inference

Having completed the training process and derived the optimal MAPLDA parameters, we can proceed with inferences on unseen data on the first $N-1$ attributes. That is, given a datum

(probe) from a test set, we aim to classify the datum into the appropriate classes for each of the corresponding $N-1$ attributes.

Since we do not have any prior knowledge of the conditions under which the data that belong to the test set may have been captured, it is very likely that the data may be perturbed by noise. Therefore, in order to determine the appropriate class, we compare the probe ($\boldsymbol{x}_p$) with a number of different data from a gallery in order to find the most likely match, in a similar manner to [9]. In essence, this boils down to maximum likelihood estimation under M (i.e., the total number of data in the gallery) different models. That is, for every model $m, m \in \{1, \ldots, M\}$, we calculate the log likelihood that the datum $\boldsymbol{x}_k$ in the gallery matches with the probe $\boldsymbol{x}_p$ and finally, we keep the pair that gives the largest log likelihood. This process falls under the so-called closed-set identification task, where a probe dataum has to be matched with a gallery datum. The algorithm can be extended to cover other scenarios such as verification or open-set identification.

Without loss of generality, let us assume a gallery with M data, all of which are labelled with different instantiations per attribute. Our aim is to find the pair that produces the maximum likelihood between the probe datum and one of the M gallery data. More formally, this corresponds to:

$$M_v = \underset{m \in \{1,\ldots,M\}}{\operatorname{argmax}} \left\{ \ln P\left(\mathcal{M}_m | \boldsymbol{X}\right) \right\} \tag{5.24}$$

where $\boldsymbol{X} \doteq \left[\boldsymbol{x}_1^T, \ldots, \boldsymbol{x}_M^T, \boldsymbol{x}_p^T\right]^T$. The optimal set of instantiations is described by the model M_v. If we consider a uniform prior for the selection of each model (i.e., $P\left(\mathcal{M}_m\right)$ is a constant for all $m \in \{1, \ldots, M\}$), then the actual log likelihood in (5.24) can calculated using Bayes' theorem as follows:

$$P\left(\mathcal{M}_m | \boldsymbol{X}\right) = \frac{P\left(\boldsymbol{X} | \mathcal{M}_m\right) P\left(\mathcal{M}_m\right)}{\sum_{m=1}^{M} P\left(\boldsymbol{X} | \mathcal{M}_m\right) P\left(\mathcal{M}_m\right)} \tag{5.25}$$

where the denominator is simply a normalising constant, ensuring the probabilities sum to 1.

Therefore, inference boils down to calculating:

$$\ln P\left(\boldsymbol{X}|\mathcal{M}_m\right) = \sum_{q=1,q\neq m}^{M} \ln P\left(\boldsymbol{x}_q\right) + \ln P\left(\boldsymbol{x}_p, \boldsymbol{x}_m\right) \tag{5.26}$$

where for each model m, the probe is paired with the m-th datum in the gallery and an individual marginal is added for the rest of the gallery data.

As aforementioned, and without loss of generality, we assume that inference is conducted for the first $N-1$ attributes. In order to perform inference without disregarding knowledge of attributes not required for inference, the sensible approach is to marginalise out the remaining N-th attribute. Then, following the process described above, we recover the optimal instantiations of attributes explained by model $\mathcal{M}_v$, utilising (5.24), (5.25) and (5.26). The joint probabilities in (5.26) are Gaussians, and therefore, they can be estimated as:

$$P\left(\boldsymbol{x}_q\right) \sim \mathcal{N}_{\boldsymbol{x}_q}\left(\mathbf{0}, \mathbf{F}\mathbf{F}^T + \mathbf{F}_N\mathbf{F}_N^T + \boldsymbol{G}\boldsymbol{G}^T + \boldsymbol{\Sigma}\right) \tag{5.27}$$

where $\mathbf{F} \doteq \begin{bmatrix} \mathbf{F}_1 & \mathbf{F}_2 & \dots & \mathbf{F}_{N-1} \end{bmatrix}$. By assigning $\boldsymbol{x}' \doteq \left[\boldsymbol{x}_p^T, \boldsymbol{x}_m^T\right]^T$ and using the "completing-the-square" method, the marginals can be estimated as:

$$P\left(\boldsymbol{x}'\right) = \mathcal{N}_{\boldsymbol{x}'}\left(\mathbf{0}, \boldsymbol{A}\boldsymbol{A}^T + \boldsymbol{\Sigma}'\right) \tag{5.28}$$

where:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{F} & \boldsymbol{G} & \mathbf{0} \\ \boldsymbol{F} & \mathbf{0} & \boldsymbol{G} \end{bmatrix}, \quad \boldsymbol{\Sigma}' = \begin{bmatrix} \boldsymbol{\Sigma} + \boldsymbol{F}_N\boldsymbol{F}_N^T & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma} + \boldsymbol{F}_N\boldsymbol{F}_N^T \end{bmatrix} \tag{5.29}$$

A graphical representation for this case can be found in Fig. 5.3.

Regarding the special case where inference about *only one* attribute is required, the marginals have the same form as in (5.27). The joint distribution, given that the attribute of interest is denoted as $i \in \{1, \dots, N\}$, follows the form:

$$P\left(\boldsymbol{x}'\right) \sim \mathcal{N}_{\boldsymbol{x}'}\left(\mathbf{0}, \boldsymbol{A}\boldsymbol{A}^T + \boldsymbol{\Sigma}'\right) \tag{5.30}$$

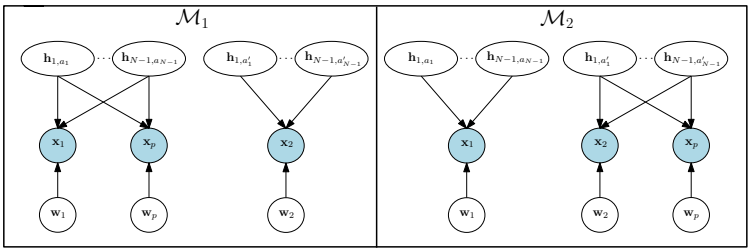


Figure 5.3: Inference for *some* attributes (in this case, the first $N-1$ attributes). For this particular case, only two data exist in the gallery, so the probe datum $\boldsymbol{x}_p$ can be matched with either datum $\boldsymbol{x}_1$ or datum $\boldsymbol{x}_2$. In case it does match with datum $\boldsymbol{x}_1$, then it is assigned labels $\{a_1, \ldots, a_{N-1}\}$ (model $\mathcal{M}_1$). Otherwise, it receives labels $\{a'_1, \ldots, a'_{N-1}\}$ (model $\mathcal{M}_2$).

where in this case:

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{F}_i & \boldsymbol{G} & \mathbf{0} \\ \\ \boldsymbol{F}_i & \mathbf{0} & \boldsymbol{G} \end{bmatrix}, \quad \boldsymbol{\Sigma}' = \begin{bmatrix} \boldsymbol{\Sigma} + \sum_{i=1, i \neq n}^{N} \boldsymbol{F}_i \boldsymbol{F}_i^T & \mathbf{0} \\ \\ \mathbf{0} & \boldsymbol{\Sigma} + \sum_{i=1, i \neq n}^{N} \boldsymbol{F}_i \boldsymbol{F}_i^T \end{bmatrix} \tag{5.31}$$

We finally note that MAPLDA is a generalisation of PLDA; in the degenerate case where only one attribute is available during training, MAPLDA reduces to PLDA.

5.3.3 3D Facial Shape Generation

We can exploit the generative property of MAPLDA, alongside the multi-attribute aspect of the model, to generate data with respect to different combinations of attribute values. Data generation can be accomplished as follows:

- Firstly, without loss of generality, we train a MAPLDA model with regards to two attributes we are interested in (e.g., attributes ethnicity and age, weight and age, etc.). After the training process, we recover the optimal $\mathbf{F}_1$, $\mathbf{F}_2$, $\mathbf{G}$ subspaces and noise diagonal covariance $\boldsymbol{\Sigma}$.

- Secondly, we pick the distinct instantiations of attributes we are interested in generating (e.g., *Chinese* ethnic group and *18-24* age group) and stack row-wise all the training

data pertaining to these instantiations, creating a new vector $\mathbf{x}'$.

- Thirdly, if $\mathbf{h}_{i,a_i}$ and $\mathbf{h}_{j,a_j}$ are the selectors corresponding to the particular attributes, we stack them row-wise, i.e., $\mathbf{h}^T \doteq \begin{bmatrix} \mathbf{h}_{i,a_i} & \mathbf{h}_{j,a_j} \end{bmatrix}$, and calculate the posterior $\mathbb{E}\left[P\left(\mathbf{h}|\mathbf{x}'\right)\right]$ as

$$\mathbb{E}\left[P\left(\mathbf{h}|\mathbf{x}'\right)\right] = \mathbf{C}\mathbf{A}^T\mathbf{D}^{-1}\mathbf{x}', \tag{5.32}$$

  where $\mathbf{A} = \begin{bmatrix} \mathbf{F}_1 & \mathbf{F}_2 \end{bmatrix}$, $\mathbf{C} = \left(\mathbf{I} + \mathbf{A}^T\mathbf{D}^{-1}\mathbf{A}\right)^{-1}$ and $\mathbf{D} = \left(\mathbf{\Sigma}' + \mathbf{G}'\mathbf{G}'^T\right)^{-1}$, where $\mathbf{\Sigma}'$ is defined as in (5.14), and $\boldsymbol{G}'$ is a block-diagonal matrix with copies of $\boldsymbol{G}$ on the diagonal.

- Finally, for selector $\mathbf{w}$, we choose a random vector from the multivariate normal distribution and the generated datum will be rendered as

$$\mathbf{x}_g = \mathbf{A}\mathbb{E}\left[P\left(\mathbf{h}|\mathbf{x}'\right)\right] + \mathbf{G}\mathbf{w}. \tag{5.33}$$

  Examples of generated shapes are provided in the next section.

5.4 Experiments

Having described the training and inference procedure for MAPLDA, in this section we demonstrate the effectiveness of MAPLDA against PLDA [9], DS-LDA [150], Ioffe's PLDA variant [8], the Bayesian approach [151], LDA [152] and PCA [41], by performing several experiments on facial shapes from MeIn3D dataset [146]. We should note here that we do not compare our algorithm against DL methods. DL methods tailored to 3D face data are still in their infancy and the state-of-the-art in 3D face anlysis tasks is still provided by subspace learning methods, such as PCA [12]. In the experiments that follow we only take into account the 3D shape of the human face *without* any texture information.

MeIn3d Dataset

MeIn3D dataset [146] consists of $10,000$ raw facial scans that describe a large variation of the population. More specifically, MeIn3D dataset [146] consists of data annotated with multiple

Table 5.1: *Ethnicity* identification. Average identification rates ± standard deviations per method. MAPLDA outperforms all of the compared methods.

Method	Mean	Std
MAPLDA	**0.990**	0.051
PLDA	0.927	0.084
DS-LDA	0.919	0.073
PLDA (Ioffe)	0.917	0.089
Bayesian	0.911	0.077
LDA	0.878	0.079
PCA	0.634	0.083

attributes (i.e., ethnicity, age, weight), thus it is highly appropriate for evaluating MAPLDA. Before performing any type of training or inference the scans are consistently re-parametrised into a form where the number of vertices, the triangulation and the anatomical meaning of each vertex are made consistent across all meshes. In this way, all the training and test meshes are brought into dense correspondence. In order to achieve this task, we employ an optimal step non-rigid ICP algorithm [153]. We utilise the full spectrum of $10,000$ meshes where each mesh is labelled for a specific identity, age and ethnicity. The training and the inference are performed directly on the vectorised re-parametrised mesh of the form $\mathbb{R}^{3*N\times 1}$, where N is the distinct number of vertices.

5.4.1 Ethnicity Identification

In this experiment, we identify the *ethnicity* attribute for a given 3D shape based on its shape features, regardless of the *age-group* attribute (i.e., by marginalising out the attribute *age-group*). We split the *ethnicity* attribute into three groups consisting of *White*, *Black* and *Asian* ethnic groups. We used 85% of the MeIn3D data for training and the rest for testing. Moreover, for each experiment, we used three random test data, with each test datum belonging in a different ethnic group. For the gallery we use the same set of distinct ethnic groups used in

Table 5.2: Confusion matrices of MAPLDA and PLDA for the *ethnicity* identification experiment. By incorporating the knowledge of the *age-group* attribute in the training phase, MAPLDA is able to better discriminate between the different ethnicities. In particular, MAPLDA classifies correctly all of the *Black* people in contrast to PLDA.

Actual	Predicted			Acc
	White	Black	Chinese	
White	**0.99**	0.00	0.01	0.99
Black	0.00	**1.00**	0.00	1.00
Chinese	0.02	0.00	**0.98**	0.98

(a) MAPLDA

Actual	Predicted			Acc
	White	Black	Chinese	
White	**0.97**	0.01	0.02	0.97
Black	0.04	**0.89**	0.07	0.89
Chinese	0.05	0.02	**0.93**	0.93

(b) PLDA [9]

test samples from three random identities. We execute a total of 100 random experiments (i.e., we repeat the aforementioned process 100 times for randomly chosen test data and galleries in every experiment). Average identification rates along with the corresponding standard deviations per setting are shown in Table 5.1. Confusion matrices for MAPLDA and PLDA are provided in Table 5.2. As can be seen, MAPLDA outperforms all of the compared methods, thus demonstrating the advantages of joint attribute modelling.

5.4.2 Age-group Identification

In this experiment, we identify the *age-group* for a given datum, regardless of the *ethnicity* attribute (i.e., by marginalising out the *ethnicity* attribute). We split the *age-group* attribute into four groups consisting of under 18 years old (*<18*), *18-24*, *24-31* and *31-60* years old

Table 5.3: *Age-group* identification. Average identification rates ± standard deviations per method. MAPLDA outperforms all of the compared methods.

Method	Mean	Std
MAPLDA	**0.695**	0.063
PLDA [9]	0.540	0.079
PLDA (Ioffe) [8]	0.534	0.068
DS-LDA [150]	0.531	0.059
Bayesian [151]	0.529	0.071
LDA [152]	0.464	0.065
PCA [41]	0.327	0.074

groups. We used 85% of the MeIn3D data for training and the rest for testing. Moreover, for each experiment we used four different random test data, with each test datum belonging in a different age group. For the gallery we use the same set of distinct age groups used in the test data from four random identities. We execute 100 random experiments per setting (i.e., we repeat the aforementioned process 100 times for randomly chosen probes and galleries in every experiment). Average identification rates along with the corresponding standard deviations per setting are shown in Table 5.3. Confusion matrices for MAPLDA and PLDA are provided in Table 5.4. The identification rates are considerably lower compared to the *ethnicity* experiment and that demonstrates that the task of inferring the age of a certain face by the shape of it is a challenging one. Nevertheless, our proposed framework exhibits performance that outperforms all of the compared methods by a large margin.

5.4.3 Weight-group Identification

In this experiment, we identify the *weight-group* attribute for a given datum, regardless of the *age-group* attribute (i.e., by marginalising out the attribute *age-group*). We split the weight attribute into five groups consisting of *30-45* kg, *45-55* kg, *55-62* kg, *62-70* kg and *70-80* kg groups. We used 85% of the MeIn3D data for training and the rest for testing. Similarly to our

Table 5.4: Confusion matrices of MAPLDA and PLDA for the *age-group* identification experiment. By incorporating the knowledge of the *ethnicity* attribute in the training phase, MAPLDA is able to better discriminate between the different age-groups.

Actual	Predicted				Acc
	< 18	18-24	24-31	31-60	
< 18	**0.77**	0.18	0.05	0	0.77
18-24	0.14	**0.62**	0.23	0.01	0.62
24-31	0.02	0.20	**0.66**	0.12	0.66
31-60	0	0.06	0.19	**0.75**	0.75

(a) MAPLDA

Actual	Predicted				Acc
	< 18	18-24	24-31	31-60	
< 18	**0.59**	0.27	0.13	0.01	0.59
18-24	0.17	**0.48**	0.31	0.04	0.48
24-31	0.02	0.24	**0.52**	0.22	0.52
31-60	0.02	0.13	0.28	**0.57**	0.57

(b) PLDA [9]

previous experiments, we use five different random test data, with each test datum belonging in a different weight group. For the gallery we use the same set of distinct weight groups used in the test samples from five random identities. We execute 100 random experiments per setting (i.e., we repeat the aforementioned process 100 times for randomly chosen test data and galleries in every experiment). Average identification rates along with the corresponding standard deviations per setting are shown in Table 5.5. Confusion matrices for MAPLDA and PLDA are provided in Table 5.6. Weight identification is considered to be the most challenging experiment of all three, as predicting the correct *weight group* solely from 3D facial shapes without considering the scaling factor is a very difficult problem. Nevertheless, as it can be seen in Table 5.5, the top performance is given by MAPLDA which is 51.6%, outperforming

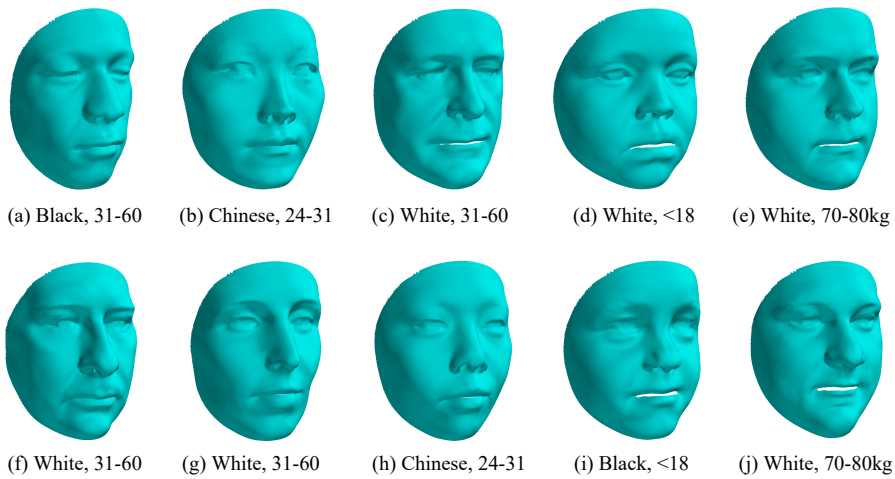


Figure 5.4: 3D facial shapes generated via MAPLDA for different attribute combinations. Figures (a-d) and (f-i) visualise different instantiations of attributes *ethnicity* and *age group*, while Figures (e,j) of attributes *ethnicity* and *weight group*.

Table 5.5: *Weight-group* identification. Average identification rates ± standard deviations per method. MAPLDA outperforms all of the compared methods.

Method	Mean	Std
MAPLDA	**0.516**	0.051
PLDA [9]	0.380	0.084
PLDA (Ioffe) [8]	0.373	0.049
DS-LDA [150]	0.368	0.054
Bayesian [151]	0.364	0.071
LDA [152]	0.346	0.059
PCA [41]	0.197	0.062

the other methods by a large margin.

5.4.4 Generating data

As thoroughly described in Section 5.3.3, the novel, multi-attribute nature of MAPLDA can be exploited to generate data with regards to a particular combination of attributes. By

Table 5.6: Confusion matrices of MAPLDA and PLDA for the *weight-group* identification experiment. By incorporating the knowledge of the *age-group* attribute in the training phase, MAPLDA is able to better discriminate between the different weight-groups.

Actual	Predicted					Acc
	30-45	45-55	55-62	62-70	70-80	
30-45	**0.55**	0.26	0.14	0.04	0.01	0.55
45-55	0.23	**0.58**	0.11	0.05	0.03	0.58
55-62	0.09	0.15	**0.46**	0.23	0.07	0.46
62-70	0.02	0.10	0.19	**0.53**	0.16	0.53
70-80	0.02	0.08	0.17	0.24	**0.49**	0.49

(a) MAPLDA

Actual	Predicted					Acc
	30-45	45-55	55-62	62-70	70-80	
30-45	**0.41**	0.31	0.19	0.06	0.03	0.41
45-55	0.26	**0.44**	0.20	0.07	0.03	0.44
55-62	0.10	0.22	**0.32**	0.28	0.08	0.32
62-70	0.04	0.12	0.25	**0.38**	0.21	0.38
70-80	0.06	0.11	0.18	0.30	**0.35**	0.35

(b) PLDA [9]

utilising MeIn3D [146] dataset, we can train a multi-attribute model with regards to e.g., the *ethnicity* and *age-group* attributes and thus generate bespoke shapes that belong in a specific combination of attribute instantiations (e.g., ethnic group *Asian* and age group *24-31*). In Fig. 5.4, we visualise some examples of generated shapes belonging to a distinct combination of attributes such as *ethnicity* and *age-group* and *ethnicity* and *weight-group*.

5.5 Conclusions

In this Chapter, we introduced Multi-Attribute PLDA, a novel CA method that is able to *jointly* model observations enriched with labels in terms of multiple attributes. We provided a probabilistic formulation and optimisation procedure for training, as well as a flexible and efficient framework for inference on any subset of the attributes available during training. Evaluation was performed via several experiments on 3D facial shapes, namely ethnicity, age, and weight identification as well as 3D face generation under arbitrary instantiations of attributes. Results showed that MAPLDA outperformed all compared methods, deeming the advantages of *joint* attribute modelling apparent.

Chapter 6

3DFaceGAN: Adversarial Nets for 3D Face Representation and Generation

Contents

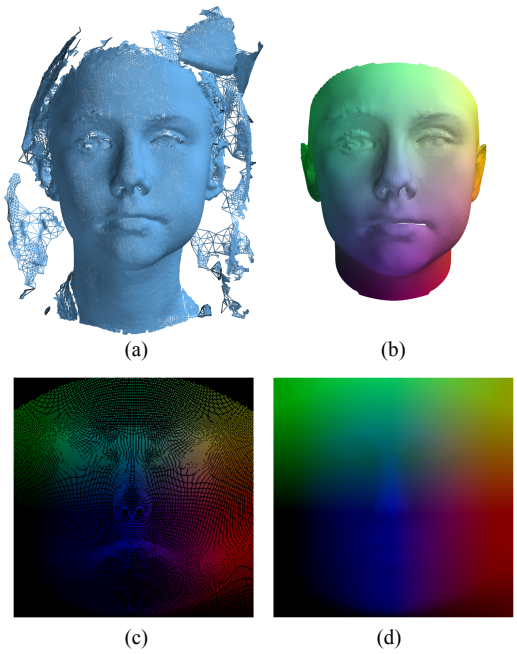


Figure 6.1: UV extraction process. In (a) we present a raw mesh, in (b) the registered mesh using the LSFM template [146], in (c) the unwrapped 3D mesh in the 2D UV space, and finally in (d) the interpolated 2D UV map. Interpolation is carried out using the 2D nearest-neighbour method.

6.1 Introduction

GANs are a promising unsupervised machine learning methodology implemented by a system of two deep neural networks competing against each other in a zero-sum game framework [20]. GANs became immediately very popular due to their unprecedented capability in terms of implicitly modelling the distribution of visual data, thus being able to generate and synthesise novel yet realistic images and videos, by preserving high-frequency details of the data distribution and hence appearing authentic to human observers. Many different GAN architectures have been proposed over the past few years, such as the Deep Convolutional GAN (DCGAN) [62] and the Progressive GAN (PGAN) [23], which was the first to show impressive results in generation of high-resolution images.

A type of GANs which has also been extensively studied in the literature is the so-called

Conditional GAN (CGAN) [34], where the inputs of the generator as well as the discriminator are conditioned on the class labels. Applications of CGANs include domain transfer [154, 155, 156], image completion [157, 158, 159], image super-resolution [160, 161, 162] and image translation [35, 163, 164, 25].

Despite the great success, GANs have had in 2D image/video generation and representation, no GAN method tailored towards tackling the aforementioned tasks in 3D shapes has been introduced in the literature. This is primarily attributed to the lack of appropriate decoder networks for meshes that are able to retain high frequency details [165, 166].

In this Chapter, we study the task of representation and generation of 3D facial surfaces using GANs. Examples of the applications of 3DFaceGAN in the tasks of 3D face representation and generation are presented in Fig. 6.2. Other DL methods that have been tried on 3D data include volumetric representations and geometric DL approaches. Nevertheless, volumetric representations lead to very low-quality representations [167, 168], and geometric DL methods [27] preserve only the low-frequency details. In 3DFaceGAN, we study approaches that use 2D convolutions in a UV unwrapping of the 3D face. The process of unwrapping a 3D face in the UV domain is shown in Fig. 6.1. Overall, key points of the Chapter can be summarised as follows.

- We introduce a novel autoencoder-like network architecture for GANs, which achieves state-of-the-art results in tasks such as 3D face representation and generation.
- We introduce a novel training framework for GANs, especially tailored for 3D facial data.
- We introduce a novel process for generating realistic 3D facial data, retaining the high frequency details of the 3D face.

The rest of the Chapter is structured as follows. In Section 6.2, we present all the details with respect to 3DFaceGAN training process, losses, and model architectures. Finally, in

Figure 6.2: 3D face representation and generation utilising the proposed 3DFaceGAN. In (a) we demonstrate the 3D face representation capability of 3DFaceGAN. The first row shows the reconstructed 3D faces whereas the second row shows the corresponding real 3D faces. As evidenced, 3DFaceGAN is able to capture and reconstruct non-linear details of the 3D face such as lips, eyelids, etc. In (b) we present the generative nature of 3DFaceGAN. The left and right hand-side show the real 3D face targets. The provided samples in between show the generations derived by the interpolations of the targets in the latent space.

Section 6.3, we provide information about the database we collected, the preprocessing we carried out in the databases we utilised for the experiments and lastly we present an extensive range of experiments of 3DFaceGAN against other state-of-the-art deep networks.

6.2 3DFaceGAN

In this Section we describe the training process, network architectures, and loss functions we utilised for 3DFaceGAN. Moreover, we discuss the framework we utilised for 3D face generation as well as present an extension of 3DFaceGAN which is able to handle data annotated with multiple labels. Finally, we should point out that while most GANs use discriminator architectures with logit outputs, in 3DFaceGAN we use an autoencoder as a discriminator for

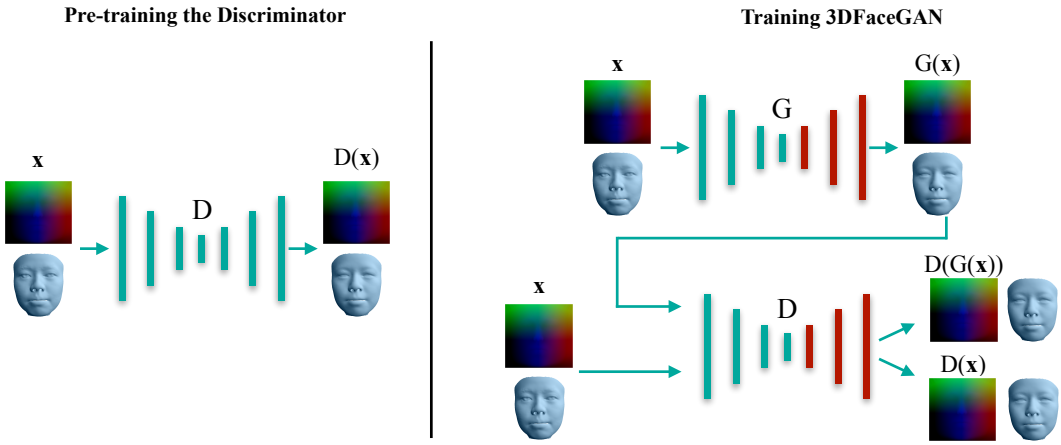


Figure 6.3: 3DFaceGAN training process overview. The networks receive (extract) 2D facial UVs as inputs (outputs). The corresponding 3D faces are shown below or next to them. We firstly pre-train D (left figure). We then use the learned weights/biases to initialise D and G and subsequently start the adversarial training (right figure). The decoder parts of D and G are depicted in red color as we freeze the weights/biases updates during the training phase of 3DFaceGAN.

reasons that are thoroughly explained in Section 6.2.2.

6.2.1 Objective function

The main objective of the generator G is to retrieve a facial UV map x as input and generate a *fake* one, $G(x)$, which in turn should be as close as possible to the *real* target facial UV map y. For example, in the case of 3D face reconstruction the input can be a 3D facial UV map and the output a reconstruction of the particular 3D facial UV map. The goal of the discriminator D is to distinguish between the *real* (y) and *fake* ($G(x)$) facial UV maps. Throughout the training process, D and G compete against each other until they reach an equilibrium, i.e., until D can no longer differentiate between the *fake* and the *real* facial UV maps.

Adversarial loss. To achieve the 3DFaceGAN objective, we propose to utilise the following

novel loss for the adversarial part. That is,

$$\begin{aligned} \mathcal{L}_D &= \mathbb{E}_y \left[\mathcal{L}\left(y\right)\right] - \lambda_{adv} \cdot \mathbb{E}_x \left[\mathcal{L}\left(G(x)\right)\right], \\ \mathcal{L}_G &= \mathbb{E}_x \left[\mathcal{L}\left(G(x)\right)\right], \end{aligned} \tag{6.1}$$

where $D\left(\cdot\right)$ refers to the output of the discriminator D, $\mathcal{L}\left(x\right) \doteq \left\|x - D(x)\right\|_1$, and λ_{adv} is the hyper-parameter which controls how much weight should be put on $\mathcal{L}\left(G(x)\right)$. The higher the λ_{adv}, the more emphasis D puts on the task of differentiating between the real and fake data. The lower the λ_{adv}, the more emphasis D puts on reconstructing the actual real data. There is a fine line between which task D should primarily focus on by adjusting λ_{adv}. In our experiments we deduced that for relatively low values of λ_{adv} we retrieve optimal performance as then D is able to influence the updates of G in such a way that the generated facial UV maps are more realistic. During the adversarial training, D tries to minimise $\mathcal{L}_D$ whereas G tries to minimise $\mathcal{L}_G$. Similar to recent works such as [67, 63], the discriminator D has the structure of an autoencoder. Nevertheless, the main differences are that (a) we do not make use of the margin m as in [67] or the equilibrium constraint as in [63], and (b) we use the autoencoder structure of the discriminator and pre-train it with the *real* UV targets prior to the adversarial training. Further details about the training procedure are presented in Section 6.2.2.

Reconstruction loss. With the utilisation of the adversarial loss (6.1), the generator G is trying to "fool" the discriminator D. Nevertheless, this does *not* guarantee that the *fake* facial UV will be close to the corresponding *real*, target one. To impose this, we use an $L1$ loss between the *fake* sample $G\left(x\right)$ and the corresponding *real* one, y, so that they are as similar as possible, as in [35]. Namely, the reconstruction loss is the following.

$$\mathcal{L}_{rec} = \mathbb{E}_x \left\|G\left(x\right) - y\right\|_1. \tag{6.2}$$

Full objective. In sum, taking into account (6.1) and (6.2), the full objective becomes

$$\begin{aligned} \mathcal{L}_D &= \mathbb{E}_y \left[\mathcal{L}\left(y\right)\right] - \lambda_{adv} \cdot \mathbb{E}_x \left[\mathcal{L}\left(G(x)\right)\right], \\ \mathcal{L}_G &= \mathbb{E}_x \left[\mathcal{L}\left(G(x)\right)\right] + \lambda_{rec} \cdot \mathcal{L}_{rec}, \end{aligned} \tag{6.3}$$

where λ_{rec} is the hyper-parameter that controls how much emphasis should be put on the reconstruction loss. Overall, the discriminator D tries to minimise $\mathcal{L}_D$ while the generator G tries to minimise $\mathcal{L}_G$.

6.2.2 Training procedure

In this Section, we first describe how we pre-train the discriminator (autoencoder) D and then provide details with respect to the adversarial training of 3DFaceGAN.

Pre-training the discriminator. The majority of GANs in the literature utilise discriminator architectures with logit outputs that correspond to a prediction on whether the input fed into the discriminator is *real* or *fake*. Recently proposed GAN variations have nevertheless taken a different approach, namely by utilising autoencoder structures as discriminators [67, 63]. Using an autoencoder structure in the discriminator D is of paramount importance in the proposed 3DFaceGAN. The benefit is twofold: (a) we can pre-train the autoencoder D acting as discriminator prior to the adversarial training, which leads to better quantitative as well as more compelling visual results [1], and (b) we are able to compute the actual UV space dense loss, as compared to simply deciding on whether the input is real or fake. As we empirically show in our experiments and ablation studies, this approach encourages the generator to produce more realistic results than other, state-of-the-art methodologies.

Adversarial training. Before starting the adversarial training, we initialise the weights

[1] note that pre-training D is not possible when the outputs are logits since there are no fake data to compare against prior to the adversarial training.

and biases[2] for both the generator G and the discriminator D utilising the learned parameters estimated after the pre-training of D (the architecture of G is identical to the architecture of D). During the training phase of 3DFaceGAN, we freeze the parameter updates in the decoder parts for both the generator G and the discriminator D. Furthermore, we utilise a low learning rate on the encoder and bottleneck parts of G and D so that overall the parameter updates are relatively close to the ones found during the pre-training of D.

Network architectures. The network architectures for both the discriminator D and the generator G are the same. In particular, each network is consisted of 2D convolutional blocks with kernel size of three, stride and padding size of one. Down-sampling is achieved by average 2D pooling with kernel and stride size of two. The convolution filters grow linearly in each down-sampling step. Up-sampling is implemented by nearest-neighbour with scale factor of two. The activation function that is primarily used is ELU [169], apart from the last layer of both D and G where Tanh is utilised instead. At the bottleneck we utilise fully connected layers and thus project the tensors to a latent vector $b \in \mathbb{R}^{N_b}$. To generate more compelling visual results, we utilised skip connections [170, 171] in the first layers of the decoder part of both the generator and the discriminator. Further details about the network architectures are provided in Table 6.1, whereas the training pipeline is depicted in Fig. 6.3.

6.2.3 3D face generation

Variational autoencoders (VAEs) [53] are widely used for generating new data using autoencoder-like structures. In this setting, VAEs add a constraint on the latent embeddings of the autoencoders that forces them to roughly follow a normal distribution. We can then generate new data by sampling a latent embedding from the normal distribution and pass it to the decoder. Nevertheless, it was empirically shown that enforcing the embeddings in the training process to follow a normal distribution leads to generators that are unable to capture high frequency

[2]for brevity in the text, we will use the term parameters to refer to the weights and the biases from this point onwards.

Table 6.1: Generator/Discriminator network architectures of 3DFaceGAN. As far as the notation is concerned, C denotes the number of input/output channels, K denotes the kernel size, S denotes the stride size, P denotes the padding size, AvgPool2D denotes average 2D pooling, UpNN denotes nearest-neighbour upsampling, and SF refers to the scaling factor size of the nearest-neighbour upsampling. CONV-BLOCK(C1, C2, K, S, P) and DECONV-BLOCK(C1, C2, K, S, P) refer to a block of two convolutions where the first is CONV(C1, C2, K, S, P) followed by an ELU [169] activation function and the second is CONV(C2, C2, K, S, P), also followed by an ELU [169] activation function.

Part	Input → Output shape	Layer information
Encoder	(h, w, 3) → (h, w, n)	CONV-(Cn, K3x3, S1, P1), ELU
	(h, w, n) → ($\frac{h}{2}$, $\frac{w}{2}$, 2n)	CONV-BLOCK-(Cn, 2n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{2}$, $\frac{w}{2}$, 2n) → ($\frac{h}{4}$, $\frac{w}{4}$, 3n)	CONV-BLOCK-(C2n, C3n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{4}$, $\frac{w}{4}$, 3n) → ($\frac{h}{8}$, $\frac{w}{8}$, 4n)	CONV-BLOCK-(C3n, C4n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{8}$, $\frac{w}{8}$, 4n) → ($\frac{h}{16}$, $\frac{w}{16}$, 5n)	CONV-BLOCK-(C4n, C5n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{16}$, $\frac{w}{16}$, 5n) → ($\frac{h}{32}$, $\frac{w}{32}$, 6n)	CONV-BLOCK-(C5n, C6n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{32}$, $\frac{w}{32}$, 6n) → ($\frac{h}{32}$, $\frac{w}{32}$, 6n)	CONV-BLOCK-(C6n, C6n, K3x3, S1, P1)
Bottleneck_1	($\frac{h}{32} \times \frac{w}{32} \times$6n) → n	Fully connected
Bottleneck_2	n → ($\frac{h}{32} \times \frac{w}{32} \times$n)	Fully connected
Decoder	($\frac{h}{32}$, $\frac{w}{32}$, n) → ($\frac{h}{16}$, $\frac{w}{16}$, n)	DECONV-BLOCK(Cn, Cn K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{16}$, $\frac{w}{16}$, n) → ($\frac{h}{8}$, $\frac{w}{8}$, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{8}$, $\frac{w}{8}$, n) → ($\frac{h}{4}$, $\frac{w}{4}$, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{8}$, $\frac{w}{8}$, n) → ($\frac{h}{4}$, $\frac{w}{4}$, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{2}$, $\frac{w}{2}$, n) → (h, w, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	(h, w, n) → (h, w, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1)
	(h, w, n) → (h, w, 3)	DECONV(Cn, C3, K3x3, S1, P1), Tanh

details [172]. To alleviate this, we propose to generate data using Algorithm 8, which better retains the generated data fidelity, as shown in Section 6.3.

6.2.4 3DFaceGAN for multi-label 3D data

Over the last few years, databases annotated with regards to multiple labels are becoming available in the scientific community. For instance, 4DFAB [173] is a publicly available 3D facial database containing data annotated with respect to multiple expressions.

Algorithm 8 3D face generation algorithm.

Step 1: Train 3DFaceGAN utilising (6.3). **Step 2:** Extract the trained G, and for all N training facial UV maps: **for** $i = 1 : N$ **do**
  Input UV map $\mathbf{x}_i$ in G. Extract the corresponding bottleneck $\mathbf{z}_i \in \mathbb{R}^{N_b \times 1}$.
end
Step 3: Concatenate column-wise all of the bottlenecks, i.e., $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N]$. **Step 4:** Extract the mean $\boldsymbol{\mu}_Z$ of $\mathbf{Z}$ and the covariance $\boldsymbol{\Sigma}_Z$ of the zero-mean $\mathbf{Z}$. **Step 5:** To generate new data, retain only the trained Bottleneck$_2$ and the Decoder part of G (see Table 6.1 for the network structures) and sample a new $\mathbf{z}_i$ (i.e., Bottleneck$_2$ input) from the multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu}_Z, \boldsymbol{\Sigma}_Z)$.

We can extend 3DFaceGAN to handle data annotated with regards to multiple labels as follows. Without any loss of generality, suppose there are three labels in the database (e.g., expressions *neutral*, *happiness* and *surprise*). We adopt the so-called one-hot representation and thus denote the existence of a particular label in a datum by 1 and the absence by 0. For example, a 3D face datum annotated with the label *happiness* will have the following label representation: $l = [0, 1, 0]$, where the first entry corresponds to the label *neutral*, the second to the label *happiness* and the third to the label *surprise*. We then choose the corresponding l we want to generate and then spatially replicate it and concatenate it in the input that is then fed to the generator. The real target has also the corresponding l spatially replicated and concatenated. Apart from this change, the rest of the training process is exactly the same as the one described in Section 6.2.2.

Finally, to generate 3D facial data with respect to a particular label, we follow the same process as the one presented in Algorithm 8, with the only difference being that we extract different pairs of ($\boldsymbol{\mu}_Z$, $\boldsymbol{\Sigma}_Z$) for every subset of the data, each corresponding to a particular label in the database. We then choose the pair ($\boldsymbol{\mu}_Z$, $\boldsymbol{\Sigma}_Z$) corresponding to the desired label and sample from this multi-variate Gaussian distribution.

6.3 Experiments

In this Section we (a) describe the databases which we used to carry out the experiments utilising 3DFaceGAN, (b) provide information with respect to the data preprocessing we conducted prior to feeding the 3D data into the network, (c) succinctly describe the baseline state-of-the-art algorithms we employed for comparisons and (d) provide quantitative as well as qualitative results on a series of experiments that demonstrate the superiority of 3DFaceGAN.

6.3.1 Databases

The 3dMD database

The 3dMD database contains approximately $5,000$ 3D facial scans captured during a special exhibition in the Science Museum, London. The data were recorded with a 3dMD face capturing system. All the subjects were recorded in neutral expression.

The 3dMD apparatus utilises a 4 camera structured light stereo system which can create 3D triangular surface meshes composed of approximately $60,000$ vertices joined into approximately $120,000$ triangles. Each subject was instructed to stay still in a fixed pose during the entire scanning process with a neutral facial expression. The frame rate for every subject was constant at 8 frames per second.

Furthermore, all $5,000$ subjects provided meta-data about themselves, including their gender, age, and ethnicity. The database covers a wide variety of age, gender (48% male, 52% female), and ethnicity (82% White, 9% Asian, 5% Mixed Heritage, 3% Black and 1% other). In all of the training tasks, 85% of the data were used for training and the rest were used for testing. The subjects in the training and testing sets are disjoint.

4DFAB database

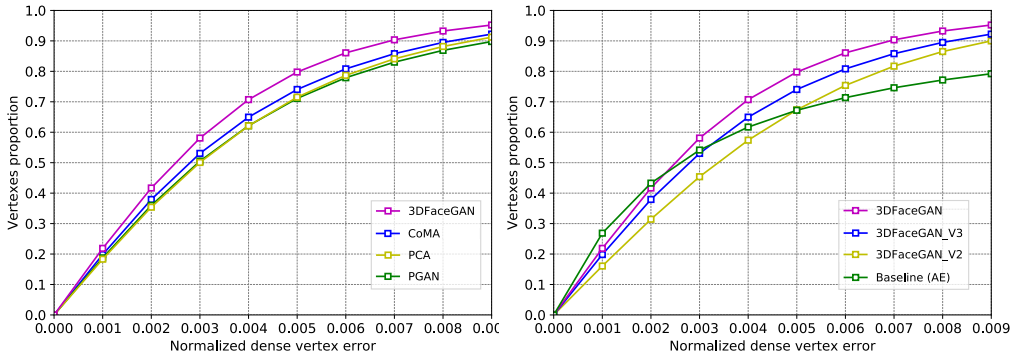
4DFAB database [173] contains 3D facial data from both male and female subjects, aged from 5 to 75 years old. The subjects vary in their ethnicity background, coming from more than 30 different ethnic groups. For the capturing process, the DI4D dynamic capturing system [3] was used.

4DFAB [173] contains data varying in expressions, such as *neutral*, *happiness*, and *surprise*. As a result, we utilised it to showcase 3DFaceGAN's capability in successfully handling data annotated with multiple labels in the task of 3D face generation. In all of the training tasks, 85% of the data were used for training and the rest were used for testing. The subjects in the training and testing sets are disjoint.

6.3.2 Data preprocessing

In order to feed the 3D data into a deep network several steps need to be carried out. Since we employ various databases, the representation of the facial topology is not consistent in terms of vertex number and triangulation. To this end, we need to find a suitable template T that can easily retain the information of all raw scans across all databases and describe them with the same triangulation/topology. We utilised the mean face mesh of the LSFM model proposed by [146], which consists of approximately $54,000$ vertices that are sufficient to capture high frequency facial details. We then bring the raw scans in dense correspondence by morphing non-rigidly the template mesh to each one of them. For this task, we utilise an optimal-step Non-rigid Iterative Closest Point algorithm [38] in combination with a per vertex weighting scheme. We weight the vertices according to the Euclidean distance measured from the tip of the nose. The greater the distance from the nose tip, the bigger the weight that is assigned to that vertex, i.e., less flexible to deform. In that way we are able to avoid the noisy information recorded by the scanners on the outer regions of the raw scans.

[3] http://www.di4d.com



(a) (b)

Figure 6.4: (a) Generalisation results on the test set for the 3D face representation task. The results are presented as cumulative error distributions of the normalised dense vertex errors. 3DFaceGAN outperforms all of the compared methods by a large margin. (b) Ablation study generalisation results for the 3D face representation task. The results are presented as cumulative error distributions of the normalised dense vertex errors.

UV maps are usually utilised to store texture information. In our case, we store the spatial location of each vertex as an RGB value in the UV space. In order to acquire the UV pixel coordinates for each vertex, we start by unwrapping our mesh template T into a 2D flat space by utilising an optimal cylindrical unwrapping technique proposed by [39]. Before storing the 3D coordinates into the UV space, all meshes are aligned in the 3D spaces by performing the General Procrustes Analysis [174] and are normalised to be in the scale of $[1, -1]$. Afterwards, we place each 3D vertex in the image plane given the respective UV pixel coordinate. Finally, after storing the original vertex coordinates, we perform a 2D nearest point interpolation in the UV domain to fill out the missing areas in order to produce a dense representation of the originally sparse UV map. Since the number of vertices in S_T is more than $50K$, we choose a $256 \times 256 \times 3$ tensor as the UV map size, which assists in retrieving a high precision point cloud with negligible re-sampling errors. A graphical representation of the preprocessing pipeline can be seen in Fig. 6.1.

Table 6.2: Generalisation metric for the meshes of the test set for the 3D face representation task. The table reports the mean error (Mean), the standard deviation (std), the Area Under the Curve (AUC), and the Failure Rate (FR) of the Cumulative Error Distributions of Fig. 6.4a.

Method	*Mean*	*std*	*AUC*	*FR (%)*
3DFaceGAN	0.0031	± 0.0028	**0.741**	**1.42e-7**
CoMA	0.0038	± 0.0037	0.716	3.66e-7
PCA	0.0040	± 0.0040	0.711	0.91e-6
PGAN	0.0041	± 0.0041	0.705	1.22e-6

Table 6.3: Ablation study generalisation results for the 3D face representation task. The table reports the Area Under the Curve (AUC) and Failure Rate (FR) of the Cumulative Error Distributions of Fig. 6.4b.

Method	*AUC*	*FR (%)*
3DFaceGAN	**0.741**	**1.42e-7**
3DFaceGAN_V3	0.736	2.62e-7
3DFaceGAN_V2	0.704	3.15e-6
Baseline (AE)	0.697	4.24-6

6.3.3 Training

We trained all 3DFaceGAN models utilising Adam [175] with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The batch size we used for the pre-training of the discriminator was 32 for a total of 300 epochs. The batch size we used for 3DFaceGAN was 16 for a total of 300 epochs. For our model we used $n = 128$ convolution filters and a bottleneck of size $b = 128$. The total number of trainable parameters was 38.5×10^6. The learning rates that we used for both the pre-training and training of the discriminator was $5e-5$ and the same was for the training of the generator. We linearly decayed the learning rate by 5% every 30 epochs during training. For the rest of the parameters, we used $\lambda_{adv} = 1e-3$, $\lambda_{rec} = 1$. Overall training time on a GV100 NVIDIA GPU was about 5 days.

Figure 6.5: Qualitative results of 3DFaceGAN compared to CoMA [29] in the 3D representation task. Moreover, heatmaps are provided, visualising the errors of both approaches against the ground truth test data. As evidenced, 3DFaceGAN is able to better capture the variation in the test data, especially in the eye and nose regions, where most of the non-linearities are present.

6.3.4 3D Face Representation

In the 3D face representation (reconstruction) experiments, we utilise the 3dMD database to train the algorithms. In particular, we feed the registered 3D data as inputs to the models and use the same data as target outputs. Before providing the qualitative as well as quantitative results, we briefly describe the baseline models we compared against as well as provide information about the error metric we used for the quantitative assessment.

Baseline models

In this Section we briefly describe the state-of-the-art models we utilised to compare 3DFaceGAN against.

Figure 6.6: Generated faces utilising 3DFaceGAN.

Vanilla Autoencoder (AE)

Vanilla Autoencoder follows exactly the same structure of the discriminator we used in 3DFaceGAN. We used the same values for the hyper-parameters and the same optimisation process. This is the main baseline we compared against and the results are provided in the ablation study in Section 6.3.4.

Convolutional Mesh Autoencoder (CoMA)

In order to train CoMA [29], we use the authors' publicly available implementation and utilise the default parameter values, the only difference being that the bottleneck size is 128, to make a fair comparison against 3DFaceGAN, where we also used a bottleneck size of 128.

Principal Component Analysis (PCA)

We employ and train a standard PCA model [176] based on the meshes of our database we used for training. We aimed at retaining the 98% of variance of our available training data which corresponds to the first 50 principal components.

Progressive GAN (PGAN)

In order to train PGAN [23], we used the authors' publicly available implementation with the default parameter values. After the training is complete, in order to represent a test 3D datum, we *invert* the generator G as in [177] and [178], i.e., we solve $z^* = \text{argmin}\,\|x - G(z)\|$ by applying gradient descent on z while retaining G fixed [178].

Error metric

A common practice when it comes to evaluating statistical shape models is to estimate the intrinsic characteristics, such as the *generalisation* of the model [179]. The *generalisation* metric captures the ability of a model to represent *unseen* 3D face shapes during the testing phase. Table 6.2 presents the generalisation metric for 3DFaceGAN compared against the baseline models. The Failure Rate mentioned in Table 6.2 and later on in the text, represents the frequency with which a method fails to represent a 3D face for a specific amount of vertices within a threshold (the bins of our graph) divided by the total number of bins. Essentially, it is the probability of failure given a threshold. Moreover, in order to compute the generalisation error for a given model, we compute the per-vertex Euclidean distance between every sample of the test set and its corresponding reconstruction. We observe that the model which holds the best error results and thus demonstrates greater generalisation capabilities is the proposed 3DFaceGAN with mean error 0.0031 and standard deviation 0.0028. Additionally, as shown in Fig. 6.4a, which depicts the cumulative error distribution of the normalised dense vertex errors, 3DFaceGAN outperforms all of the baseline models.

Ablation study

In this ablation study we investigate the importance of pre-training the discriminator D prior to the adversarial training of 3DFaceGAN as well as the freezing of the weights in the decoder parts of both D and G. More specifically, we compare 3DFaceGAN against the Vanilla Autoencoder (AE) and another two 3DFaceGAN possible variations, namely (a) the simplest

case, where the discriminator and generator structures are retained as is, but *no* pre-training takes place prior to the adversarial training (we refer to this methodology as *3DFaceGAN_V2*), (b) the case where (i) the discriminator and generator structures are retained as is, (ii) we pre-train the discriminator and initialise both the generator and the discriminator with the learned weights with *no* parameters frozen during the adversarial training (we refer to this methodology as *3DFaceGAN_V3*). As shown in Fig. 6.4b and Table 6.3, 3DFaceGAN outperforms Vanilla AE and 3DFaceGAN_V2 by a large margin. Moreover, 3DFaceGAN also outperforms 3DFaceGAN_V3. As a result, not only does 3DFaceGAN have the best performance among the compared 3DFaceGAN variants, but it also requires less training time compared to 3DFaceGAN_V3, as the parameters in the decoder parts of both the generator and the discriminator are not updated during the training phase and thus need not be computed.

6.3.5 3D Face Generation

In the 3D face generation experiment, we utilised the 3dMD database to train the algorithms. In particular, we feed the registered 3D data as inputs to the models and use the same data as target outputs.

Baseline models

The baseline models we used in this set of experiments are the same as the ones presented in Section 6.3.4.

Error metric

The metric of choice to quantitatively assess the performance of the models in this set of experiments is *specificity* [180]. For a randomly generated 3D face, *specificity* metric measures the distance of this 3D face to its nearest real 3D face belonging in the test, in terms of minimum per vertex distance over all samples of the test set. To evaluate this metric, we randomly generate $n = 10,000$ face meshes from each model. Table 6.4 reports the specificity metric

Table 6.4: Specificity metric on the test set for the 3D face generation task. We generate $10,000$ random faces from each model. The table reports the mean error (Mean) and the standard deviation (std).

Method	*Mean*	*std*
3DFaceGAN	**1.28**	**± 0.183**
CoMA	1.40	± 0.205
PCA	1.43	± 0.232
PGAN	1.79	± 0.189

Figure 6.7: Generated faces with expression utilising 3DFaceGAN multi-label approach.

for 3DFaceGAN compared against the baseline models. In order to generate random meshes utilising 3DFaceGAN, we sample from a multivariate Gaussian distribution, as explained in Section 6.2.3. To generate random meshes utilising PGAN [23], we sample new latent embeddings from the multivariate normal distribution and feed them to the generator G. To generate random faces utilising CoMA [29], we utilise the proposed variational convolutional mesh autoencoder structure, as described in [29]. For the PCA model [176], we generate meshes directly from the latent eigenspace by drawing random samples from a Gaussian distribution defined by the principal eigenvalues. As shown in Table 6.4, 3DFaceGAN achieves

the best specificity error, outperforming all compared methods by a large margin.

In Fig. 6.6, we present various visualisations of realistic 3D faces generated by 3DFaceGAN. As can be clearly seen, 3DFaceGAN is able to generate data varying in ethnicity, age, etc., thus capturing the whole population spectrum.

6.3.6 Multi-label 3D Face Generation

In this set of experiments, we utilised the 4DFAB [173] data to generate random subjects of various expressions such as *happiness* and *surprise*, as seen in Fig. 6.7. The 3D faces were generated utilising the methodology detailed in Section 6.2.4. As evinced, 3DFaceGAN is able to generate expressions of subjects varying in age and ethnicity, while retaining the high-frequency details of the 3D face.

6.4 Conclusion

In this Chapter we presented the first GAN tailored for the tasks of 3D face representation and generation. Leveraging the strengths of autoencoder-based discriminators in an adversarial framework, we presented 3DFaceGAN, a novel technique for training on large-scale 3D facial scans. As shown in an extensive series of quantitative as well as qualitative experiments against other state-of-the-art deep networks, 3DFaceGAN improves upon state-of-the-art algorithms for the tasks at hand by a significant margin.

Chapter 7

Conclusion

Contents

In this Thesis, we presented a novel multi-attribute dataset as well as a number of novel generative algorithms which are able to work efficiently in both 2D and 3D data, ranging from component analysis techniques to deep learning methodologies and in particular generative adversarial networks.

More specifically, in Chapter 3, we presented AgeDB, the first, manually collected "in-the-wild" age database, which contains annotations for multiple labels as well as attributes. AgeDB is a facial image database and comes with annotations for the attributes identity, gender, and age (where the age attribute is accurate to the year). The main advantage of AgeDB compared to other multi-attribute datasets available in the literature is that the annotations it contains are manually collected and it is thus a reliable benchmark when it comes to evaluating the performance of facial analysis algorithms. As we demonstrated in a series of quantitative and qualitative experiments in applications such as age-invariant face recognition, age estimation and progression "in-the-wild", AgeDB can be the leading standard

for the performance measuring of face analysis algorithms.

In Chapter 4, we presented a novel robust CA algorithm, tailored for 2D facial UV maps, which can be used in data that are annotated with respect to multiple attributes. The proposed Multi-Attribute Robust Component Analysis (MARCA) is able to exploit all available label information during training and it is thus able to create more informative subspaces. At the same time, because of its robustness to different types of noise such as non-Gaussian, sparse or Gaussian, it is inherently able to successfully disregard any type of noise. As a result, it outperforms other robust CA algorithms when applied on multi-attribute datasets, as we demonstrated in a number of synthetic as well as real-world applications, such as UV map denoising.

In Chapter 5, we presented Multi-Attribute Probabilistic Linear Discriminant Analysis (MAPLDA), a novel probabilistic CA algorithm tailored to 3D facial shape identification as well as generation. Based on Probabilistic Linear Discriminant Analysis (PLDA) [8, 9], which is a probabilistic model able to work under the assumption of single-attribute data, in MAPLDA we derived all the necessary theoretical frameworks so that it can work with data annotated with regards to multiple-attributes. Utilising MAPLDA on multi-attribute datasets, we are able to extract more meaningful representations of the subspaces compared to single-attribute methods such as PLDA, and thus conduct better inferences based on the derived subspaces. As we showed in a number of qualitative as well as quantitative experiments, MAPLDA is able to outperform the compared methods.

In Chapter 6, we presented 3DFaceGAN, which is a GAN tailored to representation and generation of 3D facial data. Although 3D data are non-Euclidean data and thus operators such as the 2D convolution cannot be directly applied on 3D data, in 3DFaceGAN we projec-

ted the 3D meshes on 2D UV maps and we were thus able to perform all standard operations which are typically used on 2D Euclidean data. Moreover, we developed and presented a novel GAN loss framework which is able to control the behaviour of the discriminator so that the performance of the generator is stabilised and improved. Finally, we reported a novel algorithm which can be used to generate 3D facial data with an increasing level of detail. As we showed in a number of experiments, both quantitative and qualitative, the performance of 3DFaceGAN is significantly better than the state-of-the-art.

7.1 Future Work

Although AgeDB, introduced in Chapter 3, is a very good benchmark to evaluate face analysis algorithms, several improvements can be made. Firstly, the size of AgeDB can be further extended as it now contains around 17k images. By increasing the size of the database, we can better train DL algorithms which normally requite a significant amount of data. Moreover, although AgeDB contains a different range of ethnicities, more emphasis could be put on ethnicities that are not very prevalent such as the Asian or the Black groups. Finally, more emphasis could be put in collecting data of younger or older people as most of the AgeDB identities have ages in the range of 30 to 50 years old.

One of the main arguments against DL is that we do not understand the deep features it generates and thus the behaviour of certain algorithms cannot be explained in a rigorous way. CA algorithms could step in and help in this direction and can be used alongside deep network architectures as follows: we can first extract deep features for the tasks at hand, and then feed these deep features to CA methods to conduct inferences based on the extracted subspaces. As a result, on one hand the inferences would then make more sense and on the other hand, the results could further be improved as it has been demonstrated in [181, 103]. Both MARCA and MAPLDA introduced in Chapters 4 and 5, respectively, are subspace learning algorithms.

As such, both approaches could be used on top of deep feature extraction methodologies.

MAPLDA, introduced in Chapter 5, is a probabilistic CA algorithm and use the EM algorithm in the training process. One big challenge in such probabilistic models is the initialisation of the parameters. Certain approaches resort to heuristics to make the initialisation as robust as possible [182] or others do completely random initialisations and keep the best hyperparameters found based on the validation set [11]. In order to alleviate such drawbacks, more sophisticated methods should be introduced in the literature to robustly initialise the parameters, such as the one introduced in [182]. Following this work [182], the algorithm can be extended and applied on MAPLDA.

Finally, 3DFaceGAN, introduced in Chapter 6, is one of the first attempts towards utilising GANs tailored for 3D data. Instead of first projecting the 3D data onto UV maps and then apply standard 2D convolutions, one could directly use mesh convolutions on the registered meshes. This could improve the performance of 3DFaceGAN as well as reduce the hyperparameters needed during training since UV maps contain quite a few interpolated values. Moreover, both the generator and the discriminator have an autoencoder-like structure, the latent representations could be used in order to carry out discriminative tasks such as 3D face verification.

Bibliography

[1] R. Gottumukkal and V. K. Asari, "An improved face recognition technique based on modular pca approach," *Pattern Recognition Letters*, vol. 25, no. 4, pp. 429–436, 2004. 24

[2] C. Liu, "Gabor-based kernel pca with fractional power polynomial models for face recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 5, pp. 572–581, 2004. 24

[3] J. Yang, D. Zhang, A. F. Frangi, and J.-y. Yang, "Two-dimensional pca: a new approach to appearance-based face representation and recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 1, pp. 131–137, 2004. 24

[4] M. Kaur, R. Vashisht, and N. Neeru, "Recognition of facial expressions with principal component analysis and singular value decomposition," *International Journal of Computer Applications*, vol. 9, no. 12, pp. 36–40, 2010. 24

[5] L. Oliveira, M. Mansano, A. Koerich, and A. de Souza Britto, "2d principal component analysis for face and facial-expression recognition," *Computing in Science & Engineering*, vol. 13, no. 3, pp. 9–13, 2010. 24

[6] S.-J. Wang, W.-J. Yan, G. Zhao, X. Fu, and C.-G. Zhou, "Micro-expression recognition using robust principal component analysis and local spatiotemporal directional features," in *European Conference on Computer Vision*. Springer, 2014, pp. 325–338. 24

[7] M. Scholz, F. Kaplan, C. L. Guy, J. Kopka, and J. Selbig, "Non-linear pca: a missing data approach," *Bioinformatics*, vol. 21, no. 20, pp. 3887–3895, 2005. 24

[8] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*. Springer, 2006, pp. 531–542. 24, 28, 102, 114, 117, 119, 144

[9] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8. 24, 28, 44, 102, 103, 104, 105, 111, 114, 116, 117, 118, 119, 120, 144

[10] S. Moschoglou, E. Ververas, Y. Panagakis, M. A. Nicolaou, and S. Zafeiriou, "Multi-attribute robust component analysis for facial uv maps," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 6, pp. 1324–1337, 2018. 24, 25, 29, 40, 42, 74

[11] S. Moschoglou, S. Ploumpis, M. A. Nicolaou, and S. Zafeiriou, "Multi-attribute probabilistic linear discriminant analysis for 3d facial shapes," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 493–508. 24, 25, 30, 46, 146

[12] B. Gecer, S. Ploumpis, I. Kotsia, and S. Zafeiriou, "Ganfit: Generative adversarial network fitting for high fidelity 3d face reconstruction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1155–1164. 24, 114

[13] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 3730–3738. 24, 65

[14] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in neural information processing systems*, 2014, pp. 1988–1996. 24, 65

[15] Y. Sun, X. Wang, and X. Tang, "Deep learning face representation from predicting 10,000 classes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1891–1898. 24

[16] S. Moschoglou, A. Papaioannou, C. Sagonas, J. Deng, I. Kotsia, and S. Zafeiriou, "Agedb: the first manually collected, in-the-wild age database," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 51–59. 25, 73, 92, 97, 99

[17] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015. 25, 64

[18] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1. 25, 64

[19] Y. LeCun, Y. Bengio *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995. 25

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680. 25, 28, 50, 51, 52, 53, 54, 124

[21] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986. 25, 48, 49

[22] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8110–8119. 26, 53

[23] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," *arXiv preprint arXiv:1710.10196*, 2017. 26, 53, 58, 124, 139, 141

[24] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 4401–4410. 26, 53

[25] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8798–8807. 26, 28, 125

[26] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2337–2346. 26

[27] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017. 26, 30, 49, 125

[28] G. Bouritsas, S. Bokhnyak, S. Ploumpis, M. Bronstein, and S. Zafeiriou, "Neural 3d morphable models: Spiral convolutional networks for 3d shape representation learning and generation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7213–7222. 26

[29] A. Ranjan, T. Bolkart, S. Sanyal, and M. J. Black, "Generating 3d faces using convolutional mesh autoencoders," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 704–720. 26, 30, 47, 49, 137, 138, 141

[30] S. Moschoglou, S. Ploumpis, M. Nicolaou, A. Papaioannou, and S. Zafeiriou, "3dfacegan: Adversarial nets for 3d face representation, generation, and translation," *arXiv preprint arXiv:1905.00307*, vol. 1, no. 2, 2019. 26, 50

[31] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901. 28

[32] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of eugenics*, vol. 7, no. 2, pp. 179–188, 1936. 28, 42, 43, 44

[33] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 1–37, 2011. 28, 29, 38, 40, 41, 42, 79, 84, 93, 94, 96

[34] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014. 28, 58, 125

[35] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134. 28, 60, 125, 128

[36] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017. 29, 53, 54, 55, 57

[37] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777. 29, 53, 55, 56

[38] M. De Smet and L. Van Gool, "Optimal regions for linear model-based 3d face reconstruction," in *Asian conference on computer vision*. Springer, 2010, pp. 276–289. 30, 134

[39] J. Booth and S. Zafeiriou, "Optimal uv spaces for facial morphable model construction," in *Proceedings of the IEEE International Conference on Image Processing (ICIP)*, 2014, pp. 4672–4676. 36, 135

[40] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, pp. 559–572, 1901. 37, 38, 40, 41, 43, 46

[41] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition.* IEEE Computer Society, 1991, pp. 586–587. 37, 114, 117, 119

[42] H. Abdi and L. J. Williams, "Principal component analysis," *Wiley interdisciplinary reviews: computational statistics*, vol. 2, no. 4, pp. 433–459, 2010. 37

[43] Z. Zhou, X. Li, J. Wright, E. Candes, and Y. Ma, "Stable principal component pursuit," in *2010 IEEE international symposium on information theory.* IEEE, 2010, pp. 1518–1522. 38, 40

[44] D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods.* Academic press, 2014. 40, 84

[45] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & mathematics with applications*, vol. 2, no. 1, pp. 17–40, 1976. 40, 84

[46] F. Nie, H. Huang, C. Ding, D. Luo, and H. Wang, "Robust principal component analysis with non-greedy l1-norm maximization," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 22, no. 1, 2011, p. 1433. 42, 79, 92, 93, 94, 95, 96, 97

[47] J. Ye, R. Janardan, and Q. Li, "Two-dimensional linear discriminant analysis," in *Advances in neural information processing systems*, 2005, pp. 1569–1576. 42

[48] S. Prince, P. Li, Y. Fu, U. Mohammed, and J. Elder, "Probabilistic models for inference about identity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 144–157, 2011. 44

[49] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977. 45, 105, 169

[50] F. Dellaert, "The expectation maximization algorithm," 2002. 45, 169

[51] M. A. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *AIChE journal*, vol. 37, no. 2, pp. 233–243, 1991. 46, 47

[52] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion." *Journal of machine learning research*, vol. 11, no. 12, 2010. 46, 47

[53] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013. 47, 130

[54] A. Ng *et al.*, "Sparse autoencoder," *CS294A Lecture notes*, vol. 72, no. 2011, pp. 1–19, 2011. 47, 48

[55] S. Rifai, G. Mesnil, P. Vincent, X. Muller, Y. Bengio, Y. Dauphin, and X. Glorot, "Higher order contractive auto-encoder," in *Joint European conference on machine learning and knowledge discovery in databases*. Springer, 2011, pp. 645–660. 47

[56] Y. LeCun, "The mnist database of handwritten digits," *http://yann. lecun. com/exdb/mnist/*, 1998. 48, 49, 59

[57] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in

12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16), 2016, pp. 265–283. 49

[58] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in neural information processing systems*, 2016, pp. 3844–3852. 49

[59] M. Garland and P. S. Heckbert, "Surface simplification using quadric error metrics," in *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 209–216. 49

[60] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034. 52

[61] G. Chrysos, S. Moschoglou, Y. Panagakis, and S. Zafeiriou, "Polygan: High-order polynomial generators," *arXiv preprint arXiv:1908.06571*, 2019. 53

[62] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015. 53, 124

[63] D. Berthelot, T. Schumm, and L. Metz, "Began: Boundary equilibrium generative adversarial networks," *arXiv preprint arXiv:1703.10717*, 2017. 53, 56, 57, 128, 129

[64] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802. 53

[65] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018. 53

[66] C. Villani, *Optimal transport: old and new.* Springer Science & Business Media, 2008, vol. 338. 54, 57

[67] J. Zhao, M. Mathieu, and Y. LeCun, "Energy-based generative adversarial network," *arXiv preprint arXiv:1609.03126*, 2016. 57, 128, 129

[68] G. G. Chrysos, S. Moschoglou, G. Bouritsas, Y. Panagakis, J. Deng, and S. Zafeiriou, "P-nets: Deep polynomial neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7325–7335. 58, 60

[69] W. W. Bledsoe and H. Chan, "A man-machine facial recognition system: some preliminary results," *Panoramic Research, Inc, Palo Alto, California., Technical Report PRI A*, vol. 19, p. 1965, 1965. 63

[70] G. E. Hinton, "Learning multiple layers of representation," *Trends in cognitive sciences*, vol. 11, no. 10, pp. 428–434, 2007. 64

[71] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre, "Xm2vtsdb: The extended m2vts database," in *Second international conference on audio and video-based biometric person authentication*, vol. 964, 1999, pp. 965–966. 64

[72] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker, "The cmu multi-pose, illumination, and expression (multi-pie) face database," *Technical report, Carnegie Mellon University Robotics Institute. TR-07–08*, 2007. 64

[73] ——, "Multi-pie," *Image and Vision Computing*, vol. 28, no. 5, pp. 807–813, 2010. 64, 92, 94

[74] A. M. Martinez, "The ar face database," *CVC technical report*, vol. 24, 1998. 64

[75] A. Angelova, Y. Abu-Mostafam, and P. Perona, "Pruning training sets for learning of object categories," in *Proceedings of IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, vol. 1. IEEE, 2005, pp. 494–501. 64

[76] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The feret evaluation methodology for face-recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 22, no. 10, pp. 1090–1104, 2000. 64

[77] P. J. Phillips, S. Z. Der, P. J. Rauss, and O. Z. Der, *FERET (face recognition technology) recognition algorithm development and test results.* Army Research Laboratory Adelphi, MD, 1996. 64

[78] S. A. Rizvi, P. J. Phillips, and H. Moon, "The feret verification testing protocol for face recognition algorithms," in *Proceedings of IEEE International Conference on Automatic Face and Gesture Recognition.* IEEE, 1998, pp. 48–53. 64

[79] A. Georghiades, P. Belhumeur, and D. Kriegman, "Yale face database," *Center for computational Vision and Control, Yale University*, vol. 2, 1997. 64

[80] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Technical Report 07-49, University of Massachusetts, Amherst, Tech. Rep., 2007. 65, 70

[81] E. Learned-Miller, G. B. Huang, A. RoyChowdhury, H. Li, and G. Hua, "Labeled faces in the wild: A survey," in *Advances in Face Detection and Facial Image Analysis.* Springer, 2016, pp. 189–248. 65

[82] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar, "Attribute and simile classifiers for face verification," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV).* IEEE, 2009, pp. 365–372. 65

[83] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Proceddings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE, 2011, pp. 529–534. 65

[84] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," *Proceedings of the European Conference in Computer Vision (ECCV)*, pp. 566–579, 2012. 65

[85] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in neural information processing systems*, 2014, pp. 1988–1996. 65

[86] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," *arXiv preprint arXiv:1411.7923*, 2014. 65

[87] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition." in *Proceedings of the British Machine Vision Conference (BMVC)*, vol. 1, no. 3, 2015, p. 6. 65, 71

[88] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, and A. K. Jain, "Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a," in *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1931–1939. 65

[89] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, "The megaface benchmark: 1 million faces for recognition at scale," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4873–4882. 65

[90] D. Miller, E. Brossard, S. Seitz, and I. Kemelmacher-Shlizerman, "Megaface: A million faces for recognition at scale," *arXiv preprint arXiv:1505.02108*, 2015. 65

[91] A. Lanitis and T. Cootes, "Fg-net aging data base," *Cyprus College*, 2002. 66, 67, 73, 99

[92] K. Ricanek and T. Tesafaye, "Morph: A longitudinal image database of normal adult age-progression," in *Proceedings of the IEEE International Conference on Automatic Face and Gesture Recognition*. IEEE, 2006, pp. 341–345. 66, 67

[93] Y. Fu, Y. Xu, and T. S. Huang, "Estimating human age by manifold analysis of face pictures and regression on aging features," in *Proceedings of the IEEE International Conference on Multimedia and Expo*. IEEE, 2007, pp. 1383–1386. 66, 67

[94] G. Guo, Y. Fu, C. R. Dyer, and T. S. Huang, "Image-based human age estimation by manifold learning and locally adjusted robust regression," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1178–1188, 2008. 66, 67

[95] A. C. Gallagher and T. Chen, "Understanding images of groups of people," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009, pp. 256–263. 66, 67

[96] G. Somanath, M. Rohith, and C. Kambhamettu, "Vadana: A dense dataset for facial image analysis," in *Proceedings of IEEE International Conference on Computer Vision Workshop (ICCV-W)*. IEEE, 2011, pp. 2175–2182. 66, 67

[97] E. Eidinger, R. Enbar, and T. Hassner, "Age and gender estimation of unfiltered faces," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 12, pp. 2170–2179, 2014. 66, 67

[98] G. Levi and T. Hassner, "Age and gender classification using convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR-W), Workshop on Analysis and Modeling of Faces and Gestures*, 2015, pp. 34–42. 66, 67

[99] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Cross-age reference coding for age-invariant face recognition and retrieval," in *Proceeding of the European Conference on Computer Vision*. Springer, 2014, pp. 768–783. 66, 67

[100] ——, "Face recognition and retrieval using cross-age reference coding with cross-age celebrity dataset," *IEEE Transactions on Multimedia*, vol. 17, no. 6, pp. 804–815, 2015. 66, 67

[101] R. Rothe, R. Timofte, and L. Van Gool, "Dex: Deep expectation of apparent age from a single image," in *Proceedings of the IEEE International Conference on Computer Vision, Looking at People (LAP) Workshop*, 2015, pp. 10–15. 66, 67, 72, 73

[102] ——, "Deep expectation of real and apparent age from a single image without facial landmarks," *International Journal of Computer Vision (IJCV)*, pp. 1–14, 2016. 66, 67, 72

[103] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European conference on computer vision*. Springer, 2016, pp. 499–515. 71, 145

[104] J. Deng, Y. Zhou, and S. Zafeiriou, "Marginal loss for deep face recognition," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR-W), Faces "in-the-wild" Workshop/Challenge*. IEEE, 2017. 71

[105] M. Mathias, R. Benenson, M. Pedersoli, and L. Van Gool, "Face detection without bells and whistles," in *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 720–735. 72

[106] I. Kemelmacher-Shlizerman, S. Suwajanakorn, and S. M. Seitz, "Illumination-aware age progression," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2014, pp. 3334–3341. 74, 99

[107] X. Shu, J. Tang, H. Lai, L. Liu, and S. Yan, "Personalized age progression with aging dictionary," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2015, pp. 3970–3978. 74

[108] C. Nhan Duong, K. Luu, K. Gia Quach, and T. D. Bui, "Longitudinal face modeling via temporal deep restricted boltzmann machines," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2016, pp. 5772–5780. 74, 99

[109] N. Ramanathan and R. Chellappa, "Modeling age progression in young faces," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, vol. 1. IEEE, 2006, pp. 387–394. 74

[110] W. Wang, Z. Cui, Y. Yan, J. Feng, S. Yan, X. Shu, and N. Sebe, "Recurrent face aging," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2016, pp. 2378–2386. 74

[111] C. Sagonas, Y. Panagakis, A. Leidinger, S. Zafeiriou *et al.*, "Robust joint and individual variance explained," in *Proceedings of IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2017. 74

[112] L. A. Jeni, S. Tulyakov, L. Yin, N. Sebe, and J. F. Cohn, "The first 3d face alignment in the wild (3dfaw) challenge," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2016. 78

[113] H. Kim, M. Zollhöfer, A. Tewari, J. Thies, C. Richardt, and C. Theobalt, "Inversefacenet: Deep single-shot inverse face rendering from a single image," *arXiv:1703.10956*, 2017. 78

[114] S. Zafeiriou, G. G. Chrysos, A. Roussos, E. Ververas, J. Deng, and G. Trigeorgis, "The 3d menpo facial landmark tracking challenge," in *Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCV-W)*, 2017. 78

[115] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li, "Face alignment across large poses: A 3d solution," in *Proceedings of the IEEE International Conference on Computer Vision & Pattern Recognition (CVPR)*, 2016. 78

[116] J. Booth, E. Antonakos, S. Ploumpis, G. Trigeorgis, Y. Panagakis, and S. Zafeiriou, "3d face morphable models" in-the-wild"," *arXiv preprint arXiv:1701.05360*, 2017. 78, 79, 92

[117] J. Booth and S. Zafeiriou, "Optimal uv spaces for facial morphable model construction," in *IEEE International Conference on Image Processing (ICIP)*, Oct 2014, pp. 4672–4676. 78

[118] J. Booth, A. Roussos, A. Ponniah, D. Dunaway, and S. Zafeiriou, "Large scale 3d morphable models," *International Journal of Computer Vision (IJCV)*, pp. 1–22, 2017. 78

[119] F. Shang, Y. Liu, J. Cheng, and H. Cheng, "Robust principal component analysis with missing data," in *CIKM*, 2014. 79

[120] F. Nie, J. Yuan, and H. Huang, "Optimal mean robust principal component analysis," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2014, pp. 1062–1070. 79, 92, 93, 94, 95, 96, 97

[121] Y. Panagakis, M. A. Nicolaou, S. Zafeiriou, and M. Pantic, "Robust correlated and individual component analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 38, no. 8, pp. 1665–1678, 2016. 80

[122] C. Sagonas, E. Ververas, Y. Panagakis, and S. Zafeiriou, "Recovering joint and individual components in facial data," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 40, no. 11, pp. 2668–2681, Nov 2018. 80, 84, 92, 93, 94, 95, 96, 97, 99

[123] B. Thompson, "Canonical correlation analysis," *Encyclopedia of statistics in behavioral science*, 2005. 80

[124] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics*, vol. 59, no. 6, pp. 797–829, 2006. 83

[125] P. H. Schönemann, "A generalized solution of the orthogonal procrustes problem," *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966. 86

[126] Y. Peng, A. Ganesh, J. Wright, W. Xu, and Y. Ma, "Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, no. 11, pp. 2233–2246, 2012. 87

[127] C.-T. Shen, W.-H. Lu, S.-W. Shih, and H.-Y. M. Liao, "Exemplar-based age progression prediction in children faces," in *Proceedings of the IEEE International Symposium on Multimedia*. IEEE, 2011, pp. 123–128. 99

[128] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002. 102

[129] D. L. Swets and J. J. Weng, "Using discriminant eigenfeatures for image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, no. 8, pp. 831–836, 1996. 102

[130] D. R. Hardoon, S. Szedmak, and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," *Neural computation*, vol. 16, no. 12, pp. 2639–2664, 2004. 102

[131] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 696–710, 1997. 102

[132] S. Roweis, "Em algorithms for pca and spca," *Advances in Neural Information Processing Systems*, pp. 626–632, 1998. 102

[133] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 61, no. 3, pp. 611–622, 1999. 102, 103

[134] M. A. Nicolaou, S. Zafeiriou, and M. Pantic, "A unified framework for probabilistic component analysis," in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 469–484. 102

[135] M. E. Wibowo, D. Tjondronegoro, L. Zhang, and I. Himawan, "Heteroscedastic probabilistic linear discriminant analysis for manifold learning in video-based face recognition," in *IEEE Workshop on Applications of Computer Vision (WACV)*. IEEE, 2013, pp. 46–52. 102

[136] Y. Zhang and D.-Y. Yeung, "Heteroscedastic probabilistic linear discriminant analysis with semi-supervised extension," in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2009, pp. 602–616. 102

[137] S. Yu, K. Yu, V. Tresp, H.-P. Kriegel, and M. Wu, "Supervised probabilistic principal component analysis," in *Proceedings of the International Conference on Knowledge Discovery and Data Mining*. ACM, 2006, pp. 464–473. 102

[138] A. Klami, S. Virtanen, and S. Kaski, "Bayesian canonical correlation analysis," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 965–1003, 2013. 102

[139] F. R. Bach and M. I. Jordan, "A probabilistic interpretation of canonical correlation analysis," 2005. 102

[140] C. Archambeau, N. Delannay, and M. Verleysen, "Mixtures of robust probabilistic principal component analyzers," *Neurocomputing*, vol. 71, no. 7, pp. 1274–1282, 2008. 103

[141] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural computation*, vol. 11, no. 2, pp. 443–482, 1999. 103

[142] N. Lawrence, "Probabilistic non-linear principal component analysis with gaussian process latent variable models," *The Journal of Machine Learning Research*, vol. 6, pp. 1783–1816, 2005. 103

[143] P. Li, Y. Fu, U. Mohammed, J. H. Elder, and S. J. Prince, "Probabilistic models for inference about identity," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 144–157, 2012. 103, 105

[144] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of interspeaker variability in speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 5, pp. 980–988, 2008. 104

[145] M. Lüthi, T. Gerig, C. Jud, and T. Vetter, "Gaussian process morphable models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 104

[146] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniah, and D. Dunaway, "A 3d morphable model learnt from 10,000 faces," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 5543–5552. 104, 105, 114, 120, 124, 134

[147] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, p. 248, 2015. 104

[148] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands: modeling and capturing hands and bodies together," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 6, p. 245, 2017. 104

[149] C. M. Bishop, "Pattern recognition & machine learning," *Machine Learning*, 2006. 109, 110

[150] X. Wang and X. Tang, "Dual-space linear discriminant analysis for face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2004, pp. II–II. 114, 117, 119

[151] B. Moghaddam, T. Jebara, and A. Pentland, "Bayesian face recognition," *Pattern Recognition*, vol. 33, no. 11, pp. 1771–1782, 2000. 114, 117, 119

[152] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997. 114, 117, 119

[153] B. Amberg, S. Romdhani, and T. Vetter, "Optimal step nonrigid icp algorithms for surface registration," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8. 115

[154] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 1857–1865. 125

[155] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan, "Unsupervised pixel-level domain adaptation with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 7. 125

[156] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 4. 125

[157] Y. Li, S. Liu, J. Yang, and M.-H. Yang, "Generative face completion," in *Proceedings of the the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 3. 125

[158] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 3. 125

[159] W. Wang, Q. Huang, S. You, C. Yang, and U. Neumann, "Shape inpainting using 3d generative adversarial network and recurrent convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2298–2306. 125

[160] K. Nguyen, C. Fookes, S. Sridharan, M. Tistarelli, and M. Nixon, "Super-resolution for biometrics: A comprehensive survey," *Pattern Recognition*, vol. 78, pp. 23–42, 2018. 125

[161] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 694–711. 125

[162] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. P. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, no. 3, 2017, p. 4. 125

[163] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232. 125

[164] Y. Choi, M. Choi, M. Kim, J.-W. Ha, S. Kim, and J. Choo, "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8789–8797. 125

[165] A. Dosovitskiy and T. Brox, "Generating images with perceptual similarity metrics based on deep networks," in *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 658–666. 125

[166] A. S. Jackson, A. Bulat, V. Argyriou, and G. Tzimiropoulos, "Large pose 3d face reconstruction from a single image via direct volumetric cnn regression," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1031–1039. 125

[167] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, no. 4, 2017, p. 6. 125

[168] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, p. 4, 2017. 125

[169] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," in *Proceedings of the International Conference for Learning Representations (ICLR)*, 2016. 130, 131

[170] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 770–778. 130

[171] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks." in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, no. 2, 2017, p. 3. 130

[172] O. Litany, A. Bronstein, M. Bronstein, and A. Makadia, "Deformable shape completion with graph convolutional autoencoders," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1886–1895. 131

[173] S. Cheng, I. Kotsia, M. Pantic, and S. Zafeiriou, "4dfab: A large scale 4d database for facial expression analysis and biometric applications," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2018, pp. 5117–5126. 131, 134, 142

[174] J. C. Gower, "Generalized procrustes analysis," *Psychometrika*, vol. 40, no. 1, pp. 33–51, 1975. 135

[175] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference for Learning Representations (ICLR)*, 2015. 136

[176] I. Jolliffe, "Principal component analysis," in *International Encyclopedia of Statistical Science*. Springer, 2011, pp. 1094–1096. 138, 141

[177] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, "Are gans created equal? a large-scale study," *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, 2018. 139

[178] A. Mahendran and A. Vedaldi, "Understanding deep image representations by inverting them," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 5188–5196. 139

[179] R. Davies, C. Twining, and C. Taylor, *Statistical models of shape: Optimization and evaluation*. Springer Science & Business Media, 2008. 139

[180] A. Brunton, A. Salazar, T. Bolkart, and S. Wuhrer, "Review of statistical shape spaces for 3d data with comparative analysis for human faces," *Computer Vision and Image Understanding*, vol. 128, pp. 1–17, 2014. 140

[181] L. Wan, N. Liu, H. Huo, and T. Fang, "Face recognition with convolutional neural networks and subspace learning," in *2017 2nd international conference on image, vision and computing (ICIVC)*. IEEE, 2017, pp. 228–233. 145

[182] S. Moschoglou, M. Nicolaou, Y. Panagakis, and S. Zafeiriou, "Initializing probabilistic linear discriminant analysis," in *2017 25th European Signal Processing Conference (EUSIPCO)*. IEEE, 2017, pp. 1175–1179. 146

[183] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951. 170

Appendix — Expectation Maximisation

We have mentioned and used the EM algorithm in 5, in order to compute the optimal set of parameters $\boldsymbol{\theta}$. It would be simple to estimate this set of parameters $\boldsymbol{\theta}$, if the hidden variables were known or, likewise, it would be simple to estimate the hidden variables if $\boldsymbol{\theta}$ were known. Since none of the two above options is available, we resort to the EM algorithm, which is an algorithm that can effectively manage such kind of mathematical problems.

EM-algorithm [49], [50] is in general a mathematical tool that is employed in order to derive maximum likelihood solutions to problems with hidden variables. If we denote all the visible data by $\mathbf{X}$ and the hidden variables by $\mathbf{Y}$, the likelihood may be written as follows:

$$P\left(\mathbf{X}|\boldsymbol{\theta}\right) = \sum_{\mathbf{Y}} P\left(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta}\right) \tag{7.1}$$

Our goal is to maximise (7.1). In order to achieve this, we firstly have to introduce some more notation. First of all, the joint (the visible data plus the hidden variables given the set of parameters $\boldsymbol{\theta}$) can be manipulated as follows:

$$P\left(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta}\right) = P\left(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}\right) \cdot P\left(\mathbf{X}|\boldsymbol{\theta}\right) \tag{7.2}$$

which follows from the multiplication rule of probability which states that for the dependent

variables $\mathbf{a}$ and $\mathbf{b}$, the following holds:

$$P(\mathbf{a}, \mathbf{b}) = P(\mathbf{a}|\mathbf{b}) \cdot P(\mathbf{b}) \tag{7.3}$$

As a result, from (7.2), the likelihood may be written as follows:

$$P(\mathbf{X}|\boldsymbol{\theta}) = \frac{P(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta})}{P(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})} \tag{7.4}$$

Then, for any PDF $q(\mathbf{Y})$ defined over the hidden variables and by taking the logarithm of the likelihood, the following holds:

$$\ln P(\mathbf{X}|\boldsymbol{\theta}) = \ln \frac{\frac{P(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta})}{q(\mathbf{Y})}}{\frac{P(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Y})}} \tag{7.5}$$

Moreover, since $q(\mathbf{Y})$ summed over possible values of $\mathbf{y}$ actuals to 1, (7.5) may be rewritten as follows:

$$\ln P(\mathbf{X}|\boldsymbol{\theta}) = \sum_{\mathbf{Y}} q(\mathbf{Y}) \ln P(\mathbf{X}, \boldsymbol{\theta}) = \sum_{\mathbf{Y}} q(\mathbf{Y}) \ln \frac{\frac{P(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta})}{q(\mathbf{Y})}}{\frac{P(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Y})}} \tag{7.6}$$

After some algebraic manipulation, (7.6) may be written as follows:

$$\ln P(\mathbf{X}|\boldsymbol{\theta}) = \underbrace{\sum_{\mathbf{Y}} q(\mathbf{Y}) \ln \frac{P(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta})}{q(\mathbf{Y})}}_{\mathcal{L}(q,\boldsymbol{\theta})} \underbrace{- \sum_{\mathbf{Y}} q(\mathbf{Y}) \ln \frac{P(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})}{q(\mathbf{Y})}}_{\mathrm{KL}(q||P)} \tag{7.7}$$

where $\mathcal{L}(q, \boldsymbol{\theta})$ is a functional of $q(\mathbf{Y})$ and $\mathrm{KL}(q||P)$ is the Kullback–Leibler divergence ([183]) which is a statistical tool that measures the distance between two PDFs, it is always non-negative and in our case equal to zero *iff* $P(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = q(\mathbf{Y})$. EM is an iterative process which consists of two parts: the E-step and the M-step and may be run for a couple of times until a convergence criterion is met.

E step In this step, we are trying to maximise $\mathcal{L}\left(q, \boldsymbol{\theta}^{old}\right)$ with respect to $q(\mathbf{Y})$, and at the same time we keep $\boldsymbol{\theta}^{old}$ fixed to its previous set of values (hence the exponent "*old*"). From (7.7) we can notice that $\mathcal{L}\left(q, \boldsymbol{\theta}^{old}\right) = \ln P(\mathbf{X}|\boldsymbol{\theta}) - \mathrm{KL}(q||P)$, so that means that the functional

$\mathcal{L}\left(q, \boldsymbol{\theta}^{old}\right)$, since $\ln P\left(\mathbf{X}|\boldsymbol{\theta}\right)$ is not dependent on $q\left(\mathbf{Y}\right)$, will reach its maximum value when $\mathrm{KL}\left(q||P\right)$ is minimum (so when it is equal to zero). From (7.7) we can see that $\mathrm{KL}\left(q||P\right)$ is equal to zero when $q\left(\mathbf{Y}\right) = P\left(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}\right)$. As a result, the E-step is completed by setting $q\left(\mathbf{Y}\right)$ equal to the posterior.

M step In this step, $q\left(\mathbf{Y}\right)$ is kept fixed and we maximise $\mathcal{L}\left(q, \boldsymbol{\theta}\right)$ with respect to $\boldsymbol{\theta}$. In other words, with this procedure we update the values for $\boldsymbol{\theta}$, which we now call $\boldsymbol{\theta}^{new}$. This step causes the log likelihood to further increase (unless it is already at a maximum). However, we should note that $q\left(\mathbf{Y}\right)$ has been calculated with the previous set of values for $\boldsymbol{\theta}$ (i.e. $\boldsymbol{\theta}^{old}$). That differentiates $q\left(\mathbf{Y}\right)$ from the posterior (i.e. $P\left(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}^{new}\right)$), which is calculated with the new set of values for $\boldsymbol{\theta}$, thus making the KL divergence non-zero, which results in a further increase of the log likelihood than the actual increase in the functional $\mathcal{L}\left(q, \boldsymbol{\theta}^{new}\right)$.