

Next-Generation Synthetic Face Data with Diffusion Models

Maria Vaz de Salvador
Master in Computational Statistics and Data Analysis
Department of Mathematics
Faculty of Sciences of University of Porto
2025



Next-Generation Synthetic Face Data with Diffusion Models

Maria Vaz de Salvador

Project Report carried out as part of the Master in Computational Statistics and Data Analysis

Department of Mathematics

2025

Supervisor

Ana F. Sequeira, Research Assistant at INESC TEC, Invited Assistant Professor at Faculty of Engineering of University of Porto

Co-supervisor

Hélder F. Oliveira, Assistant Professor at Faculty of Sciences of University of Porto, Senior Researcher at INESC TEC

INESCTEC

VCMI VISUAL COMPUTING & MACHINE INTELLIGENCE RESEARCH GROUP

Acknowledgements

First of all, I would like to thank my supervisor, Ana Filipa Sequeira, for her invaluable guidance, constructive feedback and constant encouragement throughout the development of this work. Her expertise and insight were essential to shaping the direction of this thesis. I would also like to thank my co-supervisor, Hélder Oliveira, whose expertise in Computer Vision was incredibly useful.

Words cannot describe my gratitude for the support and patience of Rafael Mamede, who followed my work very closely, offering advice, technical help and encouragement at every stage. His mentorship greatly improved both the quality of my research and my experience of the process.

Thank you to the INESC TEC research group for welcoming me and providing the computational resources that made the experiments possible.

Now, I will speak in Portuguese, since the following acknowledgments are personal in a way that no language short of my native one can convey.

A maior dívida de gratidão é para com os meus pais, que sempre me motivaram e apoiaram de todas as formas possíveis. Nada disto seria possível sem vocês. Gostaria também de agradecer ao meu irmão, Pedro, que sempre foi uma grande inspiração para mim e à minha avó, que me acompanhou com carinho e apoio incondicional.

Estou especialmente grata ao meu namorado, Eduardo, cuja paciência e compreensão tornaram este ano inquestionavelmente mais leve. Obrigada por acreditares sempre em mim, de uma forma tão natural que me fazia acreditar em mim mesma também.

Agradeço de forma muito especial à Leonor, que foi uma fonte constante de alegria e leveza, ajudando-me a manter-me calma e confiante sempre que surgia um problema; à Cristina e ao Paulo, que me apoiaram e motivaram imenso ao longo deste ano, estando sempre preocupados com o meu bem-estar e conforto; ao Alexandre, à Joana, e ao resto da minha família “emprestada”, por serem sempre uma fonte de aconchego e força.

Por fim, um grande obrigada aos meus amigos, Inês Rosa, Catarina Alves, Beatriz Topa, Diogo Caires, Isabella Silva e Maria Ana Sottomayor Barbosa, pela companhia e pela alegria que trouxeram a este percurso.

Resumo

O crescimento das preocupações com a privacidade e as novas regulamentações aumentaram a procura por alternativas aos dados biométricos autênticos. Esta dissertação investiga o potencial de modelos generativos baseados em difusão na geração de conjuntos de dados faciais sintéticos que simultaneamente sejam de elevada qualidade e que preservem a privacidade.

Propomos uma extensão de uma abordagem existente de geração de imagens faciais baseada em modelos de difusão latentes, treinando modelos separados por etnia com os respetivos subconjuntos de um conjunto de dados de imagens faciais. Em seguida, avaliámos os dados sintéticos obtidos com a nossa metodologia, comparando-os com dados gerados por um único modelo treinado sobre todo o conjunto de dados (referência sintética) e com um subconjunto do conjunto de dados de treino (referência real).

Definimos quatro principais questões de investigação: (i) de que forma é que o condicionamento por etnia de um modelo de difusão afeta a qualidade fotorrealista das imagens faciais sintéticas em comparação com um único modelo geral; (ii) será que o treino de modelos de difusão específicos por etnia melhora a separabilidade de identidade e o desempenho de verificação em sistemas de reconhecimento facial relativamente a um modelo único; (iii) os conjuntos de dados faciais sintéticos gerados com modelos de difusão conseguem suportar o treino de reconhecimento facial em pequena escala com um desempenho próximo do obtido com dados reais e será que o treino de modelos específicos por etnia reduz o desequilíbrio demográfico no desempenho do reconhecimento facial; e (iv) em que medida o condicionamento por etnia melhora a consistência das pistas de etnia nos embeddings gerados?

Para responder a estas questões, desenvolvemos um pipeline reprodutível que avalia: (i) a qualidade fotorrealista das imagens; (ii) a separabilidade das identidades; (iii) a utilidade para treinar um modelo de reconhecimento facial de pequena escala; e (iv) a consistência étnica.

Os dados sintéticos gerados com a nossa abordagem específica por etnia apresentaram uma qualidade fotorrealista semelhante à da referência sintética, mas não melhoraram a qualidade fotorrealista global, mantendo-se longe da distribuição do conjunto de dados real em termos de FID. Em termos de separabilidade de identidade, a nossa abordagem superou a referência sintética para todos os grupos étnicos, com a maior melhoria para

identidades africanas. No entanto, as amostras geradas com a nossa abordagem continuam longe da separabilidade de identidade apresentada pelos dados reais. Ao treinar um sistema de reconhecimento facial de pequena escala com os dados sintéticos gerados pela nossa abordagem, obtivemos precisões comparáveis às da referência sintética e da referência real. O conjunto de dados sintético gerado com a nossa abordagem superou largamente a referência sintética em termos de consistência étnica, indicando que as pistas étnicas nas imagens faciais geradas com a nossa abordagem específica por etnia correspondem de forma fiável à etnia solicitada.

Este trabalho contribui com um enquadramento prático e reprodutível para a avaliação de modelos de difusão condicionados com a etnia e mostra que esta abordagem pode aumentar significativamente a consistência dos atributos de etnia sem prejudicar a utilidade para reconhecimento facial.

Palavras-chave: modelos de difusão, dados sintéticos, reconhecimento facial, biometria com preservação da privacidade, condicionamento por etnia

Abstract

Growing privacy concerns and new regulations have increased the demand for alternatives to authentic biometric data. This thesis investigates the potential of diffusion-based generative models in generating synthetic face datasets that are both high-quality and privacy-preserving.

We extend an existing diffusion-based face generation approach by training separate models for each ethnicity on a balanced facial dataset. Synthetic samples drawn from these ethnicity-specific models are compared against data generated by a single model trained on the entire dataset (synthetic baseline) and against a subset of the original real data (real baseline).

We posed four main research questions: (i) how does conditioning a diffusion model on ethnicity affect the photorealistic quality of synthetic facial images compared to a single unconditioned model; (ii) does training ethnicity-specific diffusion models improve identity separability and verification performance in face recognition systems relative to a single unconditioned model; (iii) can synthetic face datasets produced by diffusion models support small-scale face recognition training with performance approaching that of real data and does training ethnicity-specific diffusion models reduce demographic imbalance in face recognition performance; and (iv) to what extent does ethnicity conditioning improve the consistency of ethnicity cues in generated embeddings?

To address these questions, we developed a reproducible pipeline that evaluates: (i) photorealistic image quality; (ii) identity separability; (iii) utility for training a small-scale face recognition model; and (iv) ethnicity consistency.

The synthetic data generated using our ethnicity-specific approach achieved similar photorealistic quality to the synthetic baseline but did not improve overall photorealistic scores and remained far from the real data baseline in terms of FID. In terms of identity separability, our approach outperformed the baseline for all ethnicity groups, with the largest improvement for African identities. However, the samples we generated using our approach are still far from the identity-separability presented by real data.

When training a small-scale face recognition system with the synthetic data generated by our approach, we achieved accuracies comparable to the synthetic baseline and to the real data baseline. Moreover, the synthetic dataset generated using our approach greatly outperformed the synthetic baseline in terms of ethnicity consistency, indicating

that demographic cues in the facial images generated with our ethnicity-specific approach reliably match the requested ethnicity.

This work contributes a practical and reproducible evaluation framework for diffusion models conditioned on ethnicity and shows that this approach can greatly enhance the consistency of ethnicity attributes without harming utility for face recognition.

Keywords: diffusion models, synthetic data, face recognition, privacy-preserving biometrics, ethnicity conditioning

Table of Contents

List of Tables viii
List of Figures ix
List of Abbreviations x

1. Introduction 1
2. Background Knowledge and State-of-the-art 3
   - 2.1. Generative Artificial Intelligence (GenAI) 3
     - 2.1.1. Variational Autoencoders: Probabilistic Latent-Space Modeling 6
     - 2.1.2. Generative Adversarial Networks: Adversarial Learning for Image Synthesis 8
   - 2.2. Diffusion Models: Iterative Denoising for High-Fidelity Generation 9
     - 2.2.1. Probabilistic vs. Implicit Diffusion Models 14
     - 2.2.2. Latent Diffusion Models 14
   - 2.3. Face Recognition (FR) systems 16
     - 2.3.1. Loss functions in deep FR models 20
       - 2.3.1.1. Angular softmax (SphereFace) 22
       - 2.3.1.2. Large Margin Cosine Loss (CosFace) 23
       - 2.3.1.3. Additive Angular Margin Loss (ArcFace) 24
       - 2.3.1.4. ElasticFace 25
   - 2.4. Synthetic data for Face Recognition 26
     - 2.4.1. Existing approaches for generating synthetic face data 28
     - 2.4.2. Limitations 30
   - 2.5. IDiff-Face: Latent Diffusion for Identity-Conditioned Face Synthesis 32
     - 2.5.1. Sampling 32
     - 2.5.2. Identity conditioning in IDiff-Face 33
     - 2.5.3. Advantages of IDiff-Face over GAN-based approaches 34
3. Methodology 35
   - 3.1. Overview and Goals 35
   - 3.2. Computational limitations 36
   - 3.3. Datasets for training and evaluation 36
     - 3.3.1. Racial Faces in-the-Wild (RFW) 36
     - 3.3.2. BUPT-BalancedFace 38
   - 3.4. Preprocessing: Face Alignment 39
   - 3.5. Extraction of the face embeddings 40
   - 3.6. Ethnicity classifier for the identity contexts 40
   - 3.7. Generation and classification of synthetic identity contexts 41
   - 3.8. Expanding IDiff-Face: Our Ethnicity-Specific Pipeline 42
     - 3.8.1. Baseline Approach: training a single IDiff-Face model 42
     - 3.8.2. Our Approach: training ethnicity-specific IDiff-Face models 42
   - 3.9. Training the baseline and ethnicity-specific IDiff-Face models 44
   - 3.10. Evaluation of the synthetic face data 44
     - 3.10.1. Evaluation of the ethnicity classifier 44
     - 3.10.2. Evaluation of the photorealistic image quality 45
       - 3.10.2.1. Metrics 45
       - 3.10.2.2. Protocol 48
     - 3.10.3. Evaluation of Identity Separability 49
       - 3.10.3.1. Metrics 49
       - 3.10.3.2. Protocol 50

3.10.4. Evaluation of the utility of the images for training FR 51
3.10.5. Evaluation of Ethnicity Consistency 51
4. Results and Discussion 53
4.1. Ethnicity classifier of embeddings 53
4.2. Sample quality comparison: baseline IDiff-Face vs. our ethnicity-specific extension 54
4.3. Composition of the datasets (synthetic data and real dataset used for evaluation) 55
4.4. Evaluation 56
4.4.1. Photorealistic quality 56
4.4.2. Identity separability 57
4.4.3. Utility for training FR 60
4.4.4. Ethnicity consistency 61
5. Conclusions and Future Work 64

Bibliography 66
A. Background Knowledge and State-of-the-art 79
A.1. Neural Networks 79
A.1.1. Popular activation functions 80
A.1.2. Optimization 81
A.1.3. Types of hidden layers 82
A.1.4. Feed Forward Network (FFN) 83
A.1.5. Convolutional Neural Network (CNN) 84
A.2. Stochastic processes, Markov Chains and Stationarity 85
A.3. Transformers 86
A.3.1. Multi-head attention mechanism 86
A.4. Vision Transformers (ViT) 87
A.5. Vector-Quantized Generative Adversarial Network (VQ-GAN 88
B. Methodology 90
B.1. Training 90
B.1.1. Hyper-parameters 90
B.1.2. Validation during training 91
B.2. FR Model Training and Evaluation 92
C. Results: Classifier for the contexts 94

List of Tables

3.1. Composition of the Racial Faces in-the-Wild (RFW) dataset. 37
3.2. Composition of the BUPT-BalancedFace dataset. 38
3.3. Simple overview of the metrics used to evaluate generated images. 48

4.1. Performance of the ethnicity classifier. 53
4.2. Number of images and identities of the datasets used for evaluation. 55
4.3. Comparison of photorealistic quality: ethnicity-specific approach, baseline IDiff-Face approach, real data. 56
4.4. Comparison of identity separability: ethnicity-specific approach, baseline IDiff-Face approach, real data. 59
4.5. Comparison of utility for training Face Recognition (FR): ethnicity-specific approach, baseline IDiff-Face approach, real data. 61
4.6. Comparison of ethnicity consistency: ethnicity-specific approach and baseline IDiff-Face approach. 62
4.7. Evaluation of ethnicity consistency: ethnicity-classifier performance on synthetic embeddings. 62

B.1. Default training hyper-parameters used for IDiff-Face (baseline and ethnicity-specific). 90
B.2. Hyper-parameters and settings used to train and evaluate the ResNet-18 face recognition model with ArcFace loss. 93

List of Figures

2.1. Discriminative versus Generative AI. 5
2.2. Generative models taxonomy. 6
2.3. Schematic representation of a VAE. 8
2.4. Schematic representation of a GAN. 9
2.5. Schematic representation of a Diffusion Model. 12
2.6. Scheme of a U-net architecture. 13
2.7. Schematic overview of the LDM architecture. 15
2.8. Face recognition systems: verification vs. identification (closed vs. open set). 18
2.9. Schematic of the training and inference stages in a face-recognition pipeline. 19
2.10. Illustration of the embedding space learned by a FR model. 19
2.11. Decision margins of different losses for binary classification. 25
2.12. Overview of the proposed IDiff-Face. 33

3.1. RFW: Distribution of the number of variations per identity. 37
3.2. BUPT-BalancedFace: Distribution of the number of variations per identity. . 39
3.3. Generation of random uniform context vectors. 41
3.4. Comparison between the baseline IDiff-Face approach (top) and the proposed ethnicity-specific IDiff-Face approach (bottom). 43

4.1. Examples of synthetic faces generated by each of the two approaches. . 54
4.2. Genuine and impostor score distributions for genuine pairs and impostor pairs by ethnicity and overall (our approach, baseline IDiff-Face and real data). . 58
4.3. Confusion matrices of the ethnicity classifier on synthetic data. 63

A.1. Schematic representation of a Feed Forward Network (FFN). 83
A.2. Schematic representation of a Convolutional Neural Network (CNN). . . 84
A.3. Schematic of a Vision Transformer (ViT). 88
A.4. Scheme of a Vector-Quantized Generative Adversarial Network (VQ-GAN). 89

B.1. Samples generated during training (baseline IDiff-Face model). 91
B.2. Samples generated during training (ethnicity-specific approach). . . . 92

C.1. Confusion matrix of the ethnicity classifier. 94
C.2. Training (orange) and validation (blue) losses. 95
C.3. Training (orange) and validation (blue) accuracies. 95

List of Abbreviations

AI Artificial Intelligence. 4

ANN Artificial Neural Network. 79

CA Cross-Attention. 16

CNN Convolutional Neural Network. 12, 82

CPD Contextual Partial Dropout. 32

DDPMs Denoising Diffusion Probabilistic Models. 14

EER Equal Error Rate. 49

FDR Fisher Discriminant Ratio. 49

FFN Feed-forward Network. 16

FID Fréchet Inception Distance. 45

FMR False Match Rate. 49

FNMR False Non-Match Rate. 49

FR Face Recognition. 3

GAN Generative Adversarial Network. 3– 5

GenAI Generative Artificial Intelligence. 4

IS Inception Score. 45

KL Kullback–Leibler. 7

LDM Latent Diffusion Model. 1, 3, 4

LPIPS Learned Perceptual Image Patch Similarity. 45

ML Machine Learning. 4

MLP Multi-layer perceptron. 83

MS-SSIM Multiscale Structural Similarity Index Measure. 45

SGD Stochastic Gradient Descent. 81

SOTA State-of-the-art. 3

SSIM Structural Similarity Index Measure. 45

VAE Variational AutoEncoder. 3, 4

1. Introduction

Biometric recognition is the automated process of identifying or verifying a person's identity using their unique human characteristics [1].

Face images are one of the most widely used sources of data for biometric recognition systems. Face recognition systems are used to verify or learn a person's identity, being employed in a large variety of fields such as surveillance, government facilities, airport and border control, and smartphone authentication, among others [2].

State-of-the-art deep face recognition (FR) models have traditionally relied on large-scale datasets such as CASIA-WebFace [3], MS-Celeb-1M [4] or VGGFace2 [5] to train deep neural networks. The availability of these extensive public datasets has enabled remarkable breakthroughs in the accuracy of FR models.

However, regulation changes have raised significant challenges to the use of such datasets. Article 9 of the EU's General Data Protection Regulation (GDPR) prohibits the processing of biometric data uniquely used for identifying individuals unless the subjects give explicit consent for the use of this data [6]. Additionally, Article 5(1)(e) of the AI Act, effective August 1, 2024, explicitly forbids the creation or expansion of facial recognition databases through untargeted scraping of facial images from the internet [7]. These regulations have led to the retraction or discontinuation of many large, publicly available face datasets [8].

Beyond privacy concerns, authentic face datasets also exhibit critical data imbalances. These include limited variations in pose and lighting, as well as the under-representation of protected demographic characteristics such as race and gender. Such imbalances decrease the robustness and fairness of FR models, resulting in serious ethical and legal concerns [9] [10]. These issues with authentic FR datasets have been accompanied by an increased interest in developing synthetic face datasets for FR [11].

This thesis situates itself at the intersection of **biometrics**, **privacy preservation**, and **generative AI**.

We will focus on **diffusion-based frameworks** for generating synthetic face datasets and study their ability to serve as an alternative to authentic face data. In particular, we explore pre-existing approaches [12] based on Latent Diffusion Model (LDM) [13] conditioned on identity, and propose a new approach that takes into account the ethnicity of the identities.

The central research questions we aim to answer throughout this document are:

1. How does conditioning a diffusion model on ethnicity affect the photorealistic quality of synthetic faces compared to a single general model?
2. Does training ethnicity-specific diffusion models improve identity separability relative to a single unconditioned model?
3. Can synthetic face datasets produced by diffusion models support small-scale face recognition training with performance approaching that of real data and does training ethnicity-specific diffusion models reduce demographic imbalance in face recognition performance?
4. To what extent does ethnicity conditioning improve the consistency of ethnicity cues in generated embeddings?

Contributions This work has been presented as a poster at the 18th Young Research Meeting of the University of Porto (IJUP) and as an oral communication at MECAD Open Talks 2025, contributing to the discussion on privacy-preserving biometrics and the role of generative models in shaping the future of biometric datasets.

2. Background Knowledge and State-of-the-art

This chapter provides the theoretical background used throughout this thesis and surveys the State-of-the-art (SOTA) that motivates our methodology.

We first introduce Generative AI models in contrast to the traditional discriminative modeling used for classification and regression tasks (detailed in Section 2.1). The following sections review key methodologies such as the Variational AutoEncoder (VAE) [14] and Generative Adversarial Network (GAN) [15] (Sections 2.1.1 and the 2.1.2). VAEs are presented because they introduce the notion of a probabilistic latent space and a reconstruction loss, which is used later for the latent-space modeling in Latent Diffusion Models. Then, we present GANs in a way that highlights their role as the historical predecessor of diffusion-based generative models: understanding their adversarial training and limitations, such as mode collapse, explains why diffusion models have become attractive for our application. We then proceed to discuss Diffusion Models [16], with particular attention to recent advancements such as LDM [13], which address the computational and performance challenges of traditional diffusion methods and form the basis of the framework proposed in this thesis (Section 2.2).

Next, we survey modern Face Recognition (FR) systems, highlighting the role of large-scale datasets and deep architectures in achieving high accuracy. In particular, we discuss the losses that underpin current SOTA FR, such as softmax-based margin losses (ArcFace [17], CosFace [18], SphereFace [19] and ElasticFace [20]). We then examine existing approaches for generating synthetic face data, while outlining their limitations.

Finally, we describe IDiff-Face [12], a latent-diffusion framework for identity-conditioned face synthesis that makes up the baseline for our proposed method.

2.1. Generative Artificial Intelligence (GenAI): Foundations of Modern Image Generation

Since ancient times, humans have created countless stories and myths of artificial beings that were given intelligence or even consciousness by master inventors. This historical folklore reflects a long-term ambition, mixed with a pinch of reasonable fear, that humans had to create intelligent machines for their own benefit.

The field of Artificial Intelligence (AI) is, nowadays, a field of key importance in Computer Science. Its focus lies on developing systems that are able to perform tasks that would usually require human intelligence, such as reasoning, problem-solving, perception and decision-making.

One of the most popular fields of study in AI, Machine Learning (ML), coined by Arthur Samuel in 1959 [21], relies on statistical models and data-driven learning. Unlike traditional rule-based programming, ML systems improve by identifying patterns in large datasets. In summary, the more and the better quality data these systems are fed, the better they perform. Note that an AI system does not necessarily need to use ML algorithms: for instance, a chess program following predefined rules does not "learn" from data. That being said, most modern AI applications, such as image recognition or natural language processing, depend heavily on ML. In the 2020s, with the rise of Generative Artificial Intelligence (GenAI), we are able to create original content, including text and images, rather than simply classifying or predicting outcomes. Two primary approaches stand out when it comes to handling data: **Generative AI** and **Discriminative AI**.

Discriminative AI models focus on distinguishing between different categories (labels) by modeling the conditional probability $P(Y \mid X = x)$ of the target variable Y given an observation x of the explanatory variable X, i.e., the decision boundary between classes directly, consequently making Discriminative AI methodologies incredibly useful for classification and regression tasks.

Generative AI models, on the other hand, focus on learning the underlying patterns in the data by modeling the joint probability distribution $P(X, Y)$ of a given dependent variable X and target variable Y. These models are able to *generate* new data instances x of X (text, speech, images), and, using Bayes' theorem ($P(Y \mid X = x) = \frac{P(X=x,Y)}{P(X=x)}$), can also be used for classification.

The AI field was revolutionized by the development of GenAI models, particularly the Diffusion Model and the LDM. Additionally, generative models such as the VAE and the GAN have played a crucial role in advancing generative capabilities, with GANs excelling at high-quality image synthesis through adversarial training and VAEs enabling structured latent space representations for tasks like interpolation and anomaly detection.

Next, we will present the types of generative models, their key differences, as well as some examples. For more context on Neural Networks, read Appendix A.

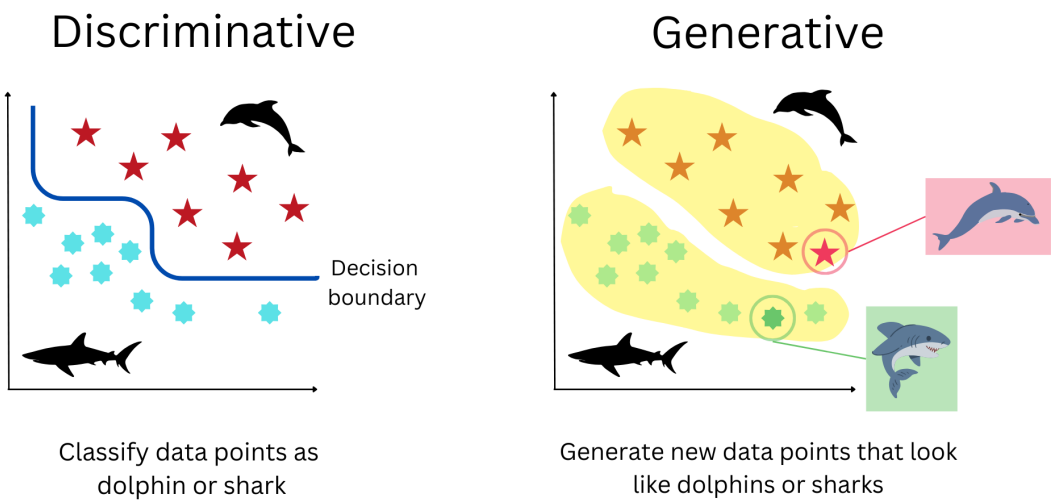


Figure 2.1: Discriminative models focus on learning decision boundaries between classes by modeling the conditional probability $P(Y|X)$. On the other hand, generative models try to capture the joint probability distribution $P(X, Y)$, allowing them not only to classify but also to generate new, realistic data samples.

Explicit density models. Explicit models model the probability distribution of the data explicitly: they learn the probability density function $p(x)$ directly, allowing the *direct computation of likelihoods* (probabilities assigned to data samples). These types of models are trained by *maximizing the likelihood of observed data*, and are able to evaluate how likely a given sample is under the learned distribution, making them especially useful for tasks requiring probability estimates, such as anomaly detection. However, they often require tractable probability distributions and are *computationally expensive for high dimensional data*, like images. Autoregressive models (such as PixelCNN [22] and WaveNet [23]), VAEs [14], Normalizing Flows [24] and Gaussian Mixture Models [25] are all examples of explicit density generative models.

Implicit models. Implicit models, instead of modeling the probability distribution explicitly, generate samples through a stochastic process, focusing on matching the data distribution indirectly. They are trained by comparing generated samples to real data and cannot evaluate $p(x)$ directly. Implicit models are able to generate high-quality and realistic samples, and are flexible and scalable for complex, high-dimensional data. However, they often present training instability, and are hard to evaluate due to lack of likelihoods [26]. A GAN is a good example of an implicit generative model [15].

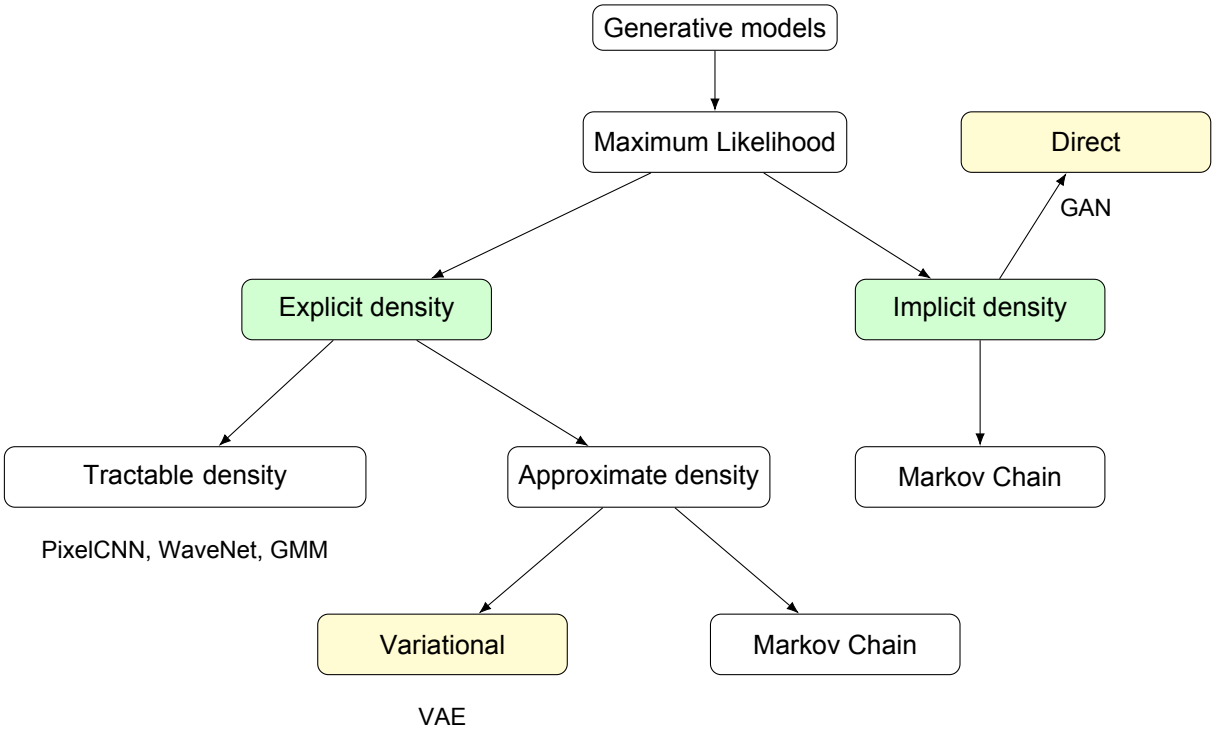


Figure 2.2: Structure of generative models according to how they perform probability density estimation.

Hybrid approaches. Modern approaches often blend explicit and implicit modeling:

- VAE-GAN hybrids combine VAEs (explicit) with GANs (implicit) for better sample quality and likelihood estimation [27].
- Score-based Models use gradients of log-density (explicit) to guide implicit sampling [28].
- Diffusion Models [16] explicitly model the data corruption process but implicitly learn to reverse it.

2.1.1 Variational Autoencoders: Probabilistic Latent-Space Modeling

The VAE is an explicit density generative model proposed in 2013 by Kingma and Welling [14], combining ideas from deep learning and probabilistic modeling. VAEs are used for multiple tasks such as image generation, anomaly detection and feature learning. They consist of two main components: an Encoder to map input image x to its latent representation z (a probability distribution, usually Gaussian), and a Decoder to reconstruct the probability distribution of x from z.

The goal of a VAE is to model the underlying probability distribution of the data. We assume the data x is generated from a latent variable z using a decoder with a conditional probability $p(x \mid z)$, where z is sampled from a well-known distribution, usually $p(z) = N(0, I)$ (standard normal distribution). The goal is to learn

$$p(x) = \int p(x|z)p(z)dz \tag{2.1}$$

The integral above is intractable, so VAEs approximate it using a variational distribution $q(z|x)$, which is parameterized by the encoder. The VAE learns this approximation by minimizing the Kullback–Leibler (KL) divergence [29] between $q(z|x)$ and the true posterior $p(z|x)$, denoted $KL(q(z|x)||p(z|x))$. The KL divergence measures how much the model probability distribution $q(z|x)$ differs from the true posterior distribution $p(z|x)$ (Equation 2.2).

$$KL(q(z|x)||p(z|x)) = \int q(z|x) \log \frac{q(z|x)}{p(z|x)} dx \tag{2.2}$$

Simplifying the previous equation, we see that the training goal is equivalent to maximizing the Evidence Lower Bound, $ELBO$ (Equation 2.3).

$$ELBO = \mathbb{E}_{q(z|x)}[\log p(x|z)] - KL(q(z|x)||p(z)) \tag{2.3}$$

The first term $\mathbb{E}_{q(z|x)}[\log p(x|z)]$ represents the reconstruction likelihood, whereas the term $KL(q(z|x)||p(z))$ ensures that our learned distribution q is similar to the true prior distribution p. Due to the latter, nearby points in the latent space produce similar reconstructions, making interpolation meaningful (smooth latent space). VAEs can also encode data into a lower-dimensional latent space while preserving its key characteristics.

Training the Encoder and the Decoder jointly minimizes the reconstruction error and the KL divergence, encouraging the latent space to follow a prior distribution (usually Gaussian) and enabling smooth interpolation and generation of new samples.

However, these models often generate blurry images, due to the fact that the reconstruction loss assumes a Gaussian likelihood, which tends to average out pixel values. Moreover, the model can struggle to capture complex, multi-modal distributions or highly nonlinear relationships in the data due to the imposed Gaussian prior. Furthermore, the trade

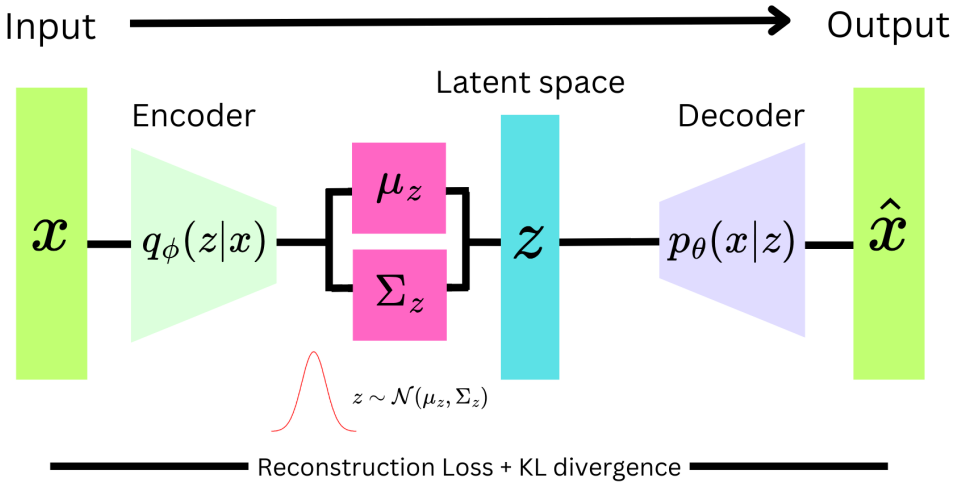


Figure 2.3: Schematic representation of a VAE. The encoder maps the input image x into a latent distribution $q(z|x)$, typically parameterized by a mean and variance. A latent variable z is sampled from this distribution and passed through the decoder to reconstruct the original data.

off between reconstruction quality and regularization in the $ELBO$ can lead to poor reconstructions if the KL term dominates, or even ignoring the latent variable z entirely (posterior collapse) if the decoder becomes too powerful.

2.1.2 Generative Adversarial Networks: Adversarial Learning for Image Synthesis

GANs were introduced in 2014 by Goodfellow et al. [15]. In a GAN, two neural networks play against one another in the form of a zero-sum game, in which one network's gain is the other's loss. The two adversarial networks consist of a generator that learns how to generate plausible data, and a discriminator that learns to distinguish real data from the generator's fake data. Basically, D and G play a two-player minimax game with objective function $V(G, D)$:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.4}$$

In practice, equation 2.4 may be unable to provide a sufficient gradient for the generator G to learn well. In early stages of training, when G still performs poorly, the discriminator D is able to reject samples with high confidence since they are clearly different from the training data. Therefore, rather than training G to minimize $\log(1 - D(G(z)))$, we train G to maximize $\log D(G(z))$, since this objective function provides much stronger gradients early in learning.

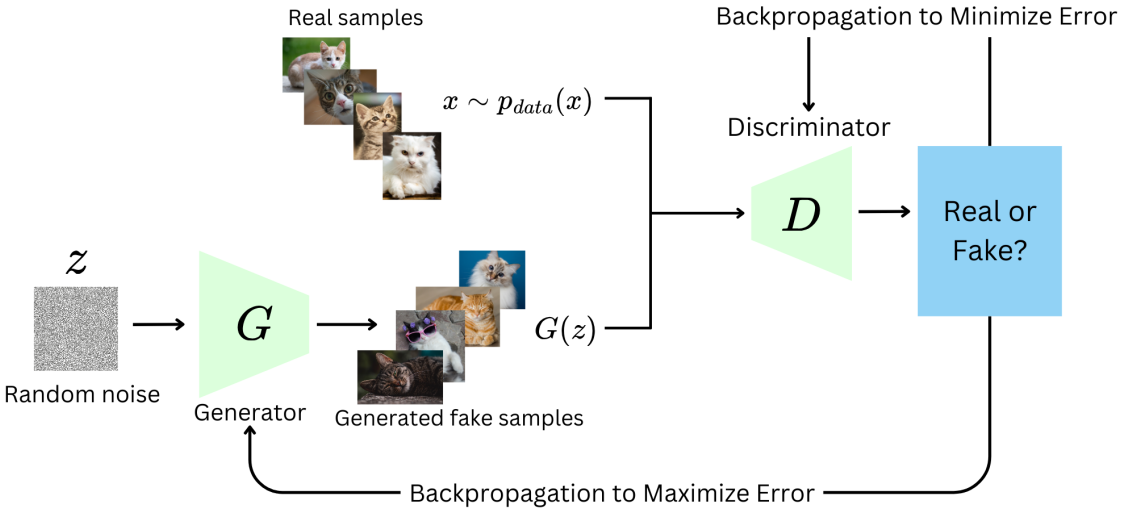


Figure 2.4: Schematic representation of a GAN. The generator G maps a random noise vector z to a synthetic sample, while the discriminator D learns to distinguish between real and generated data. The two networks are trained simultaneously in a minimax game: G aims to fool D by producing realistic samples, while D improves at detecting fakes. This adversarial process encourages the generator to produce increasingly photorealistic data.

While GANs are able to generate very realistic images, they also face issues such as mode collapse. This happens when the generator focuses on producing a limited set of data patterns that deceive the discriminator, failing to fully capture the diversity of the training data distribution, ending up producing repetitive output data [30].

2.2. Diffusion Models: Iterative Denoising for High-Fidelity Generation

Diffusion models, introduced in 2015 by Sohl-Dickstein et al. [16], are a class of generative models inspired by thermodynamics, specifically the diffusion process, which describes the movement of particles from regions of higher density to regions of lower density. In statistics this is analogous to gradually transforming a complex data distribution p_{data} on $\mathbb{R}^d$ into a simple, known prior distribution p_{prior} on the same space. We can view this as seeking a sequence of transformations ϕ_t such that

$$x_0 \sim p_{\text{data}} \quad \Longrightarrow \quad x_T = \phi_T \circ \cdots \circ \phi_1(x_0) \sim p_{\text{prior}} \tag{2.5}$$

The formal way to find the specific pair $\{\phi, p_{prior}\}$ that satisfies equation 2.5 is to use a Markov Chain and its stationary distribution (in Appendix A.2, you can read more on

Markov Chains and stationarity): by repeating the application of a transition kernel $q(x \mid x')$ whose stationary distribution is p_{prior}:

$$p_{\text{prior}}(x) = \int q(x \mid x')\, p_{\text{prior}}(x')\, dx' \tag{2.6}$$

Let the transformation ϕ on equation 2.5 be a repeated application of the transition kernel $q(x \mid x')$ over (discrete) time t.

$$x_t \sim q(x \mid x' = x_{t-1}), \qquad t = 1, \ldots, T \tag{2.7}$$

From the stationary distribution we have $x_\infty \sim p_{prior}$. In practice, we iterate for a sufficiently large $t = T$. Therefore, there exists an iterative way to go from a complex unknown distribution to a known simpler prior, and a simple and tractable choice for the pair $\{q, p_{prior}\}$, known as the Gaussian Diffusion, is given by Equation 2.8.

$$q(x_t \mid x_{t-1}) = \mathcal{N}\Big(x_t;\, \sqrt{1-\beta_t}\, x_{t-1},\, \beta_t I\Big), \quad t = 1, \ldots, T \tag{2.8}$$

$$q(x_T) = p_{\text{prior}}(x_T) = \mathcal{N}\Big(x_T; 0, I\Big) \tag{2.9}$$

Here, $\beta_t \in \mathbb{R}$ is a predefined decaying schedule. Now that we have built a diffusion process independent from the data distribution (for sufficiently large $t = T$), we only need to be able to sample from it. This leads to the forward diffusion process

$$x_0 \sim p_{\text{data}} \quad \Longrightarrow \quad x_T = \phi_T \circ \cdots \circ \phi_1(x_0) \sim \mathcal{N}(0, I) \tag{2.10}$$

Now that we are able to deconstruct the data and turn it into an almost structureless Gaussian, we need to also have a way of doing the opposite. From the theory of Markov Chains, we get that there exists a reverse diffusion process $\phi^{-1} = \{\phi_t^{-1}\}_{t=1}^{T}$ such that

$$x_T \sim \mathcal{N}(0, I) \quad \Longrightarrow \quad x_0 = \phi_1^{-1} \circ \cdots \circ \phi_T^{-1}(x_T) \sim p_{\text{data}} \tag{2.11}$$

In practice this reverse process is unknown and must be approximated. A parametric model of the reverse diffusion, denoted ϕ_θ^{-1}, is learned from data so that iteratively applying ϕ_θ^{-1} to Gaussian noise yields samples from p_{data}.

In 2021, Song et al. [28] proposed to use infinitely many noise scales (or time-steps), and achieved this by using Stochastic Differential Equations (SDEs). Here, in the forward process, data is progressively corrupted by noise. This process can be described by the following SDE:

$$\mathrm{d}X_t = -\frac{1}{2}g(t)X_t\mathrm{d}t + \sqrt{g(t)}\mathrm{d}W_t \ , g(t) > 0 \tag{2.12}$$

where the initial data point follows the distribution of the data ($X_0 \sim P_{data}$), W_t, with $t \geq 0$ is a standard Wiener process and $g(t)$ is a non-decreasing weight function. Let the marginal distribution of X_t be denoted as P_t. After an infinitesimal time dt, the forward process described by equation 2.12 decreases the magnitude of X_t and corrupts the data by adding Gaussian noise. For example, given X_0, $X_t|X_0 \sim N(\alpha(t)X_0, h(t)I_D)$, with $\alpha_t = \exp - \int_0^t \frac{1}{2}g(z)\mathrm{d}z$ and $h(t) = 1 - \alpha^2(t)$. Then, the process 2.12 transforms the data distribution P_{data} to $P_\infty = N(0, I_D)$ (variance preserving forward SDE).

The forward process 2.12 will end at sufficiently large time T > 0, with the corrupted marginal distribution $P_T \dot{\sim} N(0,1)$. The reverse diffusion process defines how diffusion models generate synthetic data. The backwards SDE is given by equation 2.13.

$$\mathrm{d}X_t^{\leftarrow} = \left[\frac{1}{2}X_t^{\leftarrow} + \nabla \log p_{T-t}(X_t^{\leftarrow})\right]\mathrm{d}t + \mathrm{d}\overline{W}_t, t \in [t, T) \tag{2.13}$$

, where $\nabla \log p_t(\dot{})$ is the **score function** (gradient of the log probability density function of P_t) and $\overline{W}_t$ is a Wiener process independent of W_t. Under certain conditions, when initialized at $X_0^{\leftarrow} \sim P_T$, the backward process $(X_t^{\leftarrow})_{0 \leq t < T}$ has the same distribution as the forward process$(X_{T-t})_{0 \leq t < T}$.

Since both the score function and the distribution P_T are unknown, in practice we don't work with equation 2.13- we substitute the unknown distribution P_T by the Gaussian distribution $N(0, I_D)$ and use the estimator to the real score function $\hat{s}(x, t)$. The estimated $\hat{s}$ is usually parametrized using a deep neural network. Substituting by these in 2.13, we get the practical backward SDE in equation 2.14.

$$\mathrm{d}X_t^{\leftarrow} = \left[\frac{1}{2}X_t^{\leftarrow} + \hat{s}(X_t^{\leftarrow}, T-t)\right]\mathrm{d}t + \mathrm{d}\overline{W}_t, X_0^{\leftarrow} \sim N(0, I_D) \tag{2.14}$$

Diffusion models generate data by simulating discrete values of 2.14 using proper step

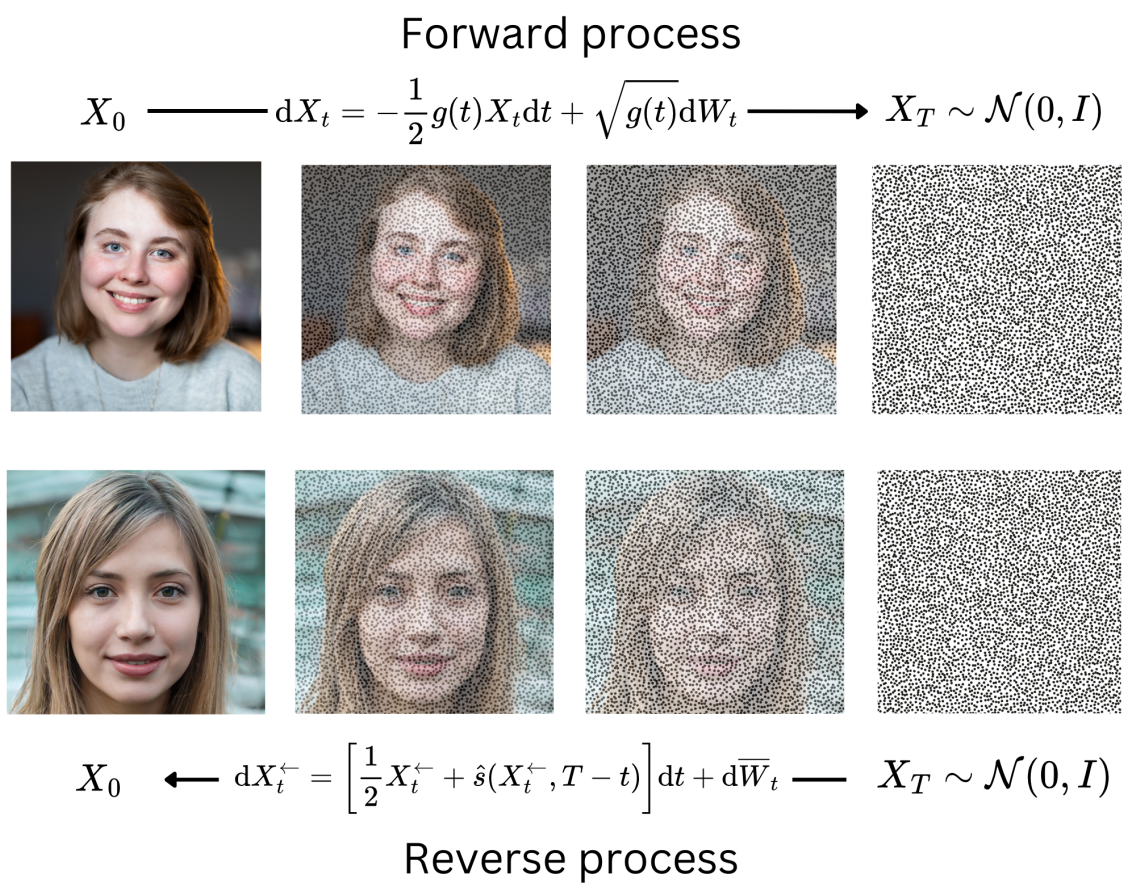


Figure 2.5: Schematic representation of a Diffusion Model. In the forward process, an input image X_0 is gradually corrupted with Gaussian noise over multiple timesteps until it becomes nearly pure noise X_T. The reverse process is then learned by a neural network (typically a U-Net), which iteratively denoises X_T step by step to recover a sample from the data distribution. By learning to approximate the reverse diffusion dynamics, the model can generate new photorealistic images starting from random noise.

size (generally use the step size order $\mathcal{O}(1/1000)$). Since traditional DMs typically model directly in pixel space, their optimization can take days to run on GPU memory. Moreover, due to the recursive nature of the reverse diffusion process, inference using these models is computationally expensive.

U-Net The U-Net, a Convolutional Neural Network (CNN) (for more details, read Appendix A.1.5), initially developed for image segmentation tasks in 2015 by Ronneberger et al. [31], is now greatly used for iterative image denoising in diffusion models. Its name is due to its U-shaped encoder–decoder architecture (Figure 2.6).

The same properties that make the U-Net effective for segmentation, such as dense, per-pixel output, multiscale context capture, and skip connections that preserve fine details,

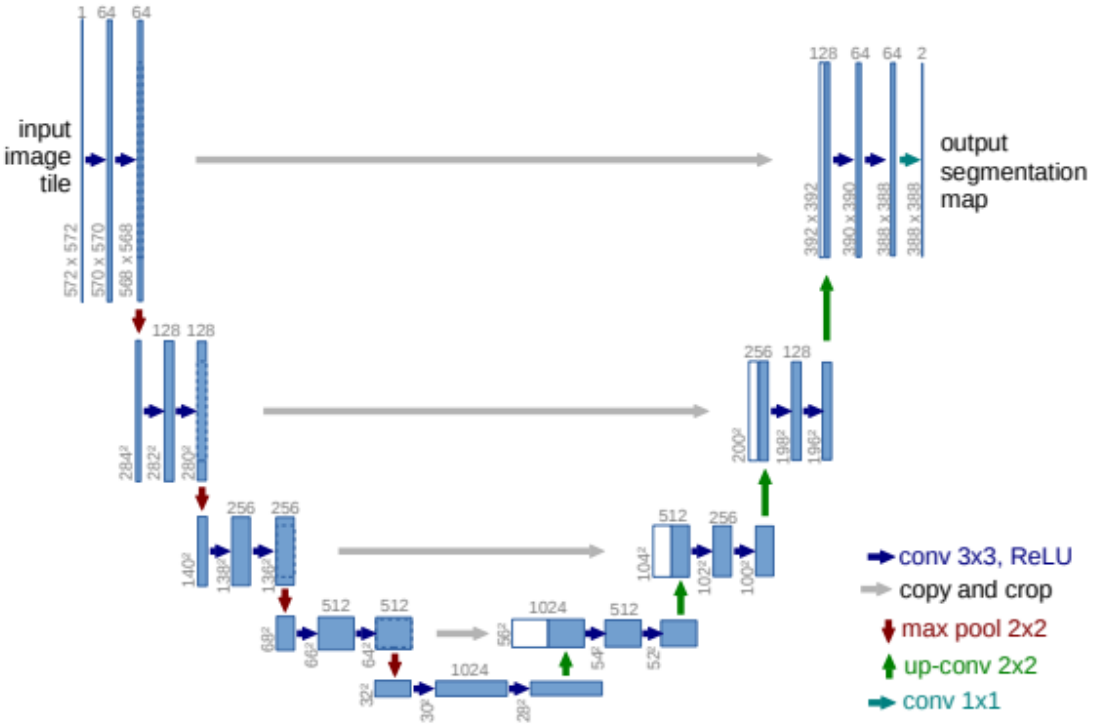


Figure 2.6: Scheme of a U-net architecture from the original paper [31]. The network follows an encoder–decoder architecture: the left side, or contracting path, applies successive convolutions and pooling operations to the input image in order to capture contextual information while reducing spatial resolution. The right side, or expanding path, uses upsampling and convolutions to recover spatial resolution. Skip connections between corresponding layers of the encoder and the decoder paths allow the U-Net to combine low-level spatial information with high-level contextual features, enabling precise localization within the image.

also make it well suited to the denoising step in diffusion models. Recall that, in these models, a network is asked, at each timestep, to predict the clean image or the added noise from a noisy input of the same size, which is a pixel-wise regression problem very similar to segmentation. The encoder–decoder with skip connections allows the model to use global context while producing sharp local predictions, which is crucial for high-quality image synthesis.

- **Contracting path (Encoder).** The contracting path consists of a typical CNN: it performs convolutions and follows them with a ReLU and a max pooling operation, as described in Figure A.2. The goal is to progressively reduce spatial dimensions while increasing the number of feature channels, and the output consists of the smallest, most compressed representation of the image.

- **Expanding path (Decoder).** In order to restore any spatial information that may

have been lost in the encoder, the expanding path upsamples the resulting feature maps and produces a relevant segmentation map through a sequence of up-convolutions and concatenations with high-resolution features resulting from the contracting path.

2.2.1 Probabilistic vs. Implicit Diffusion Models: A Comparative Overview

The original family of denoising diffusion models introduced by Sohl–Dickstein et al. [16] and popularized by Ho et al. [32] are commonly referred to as Denoising Diffusion Probabilistic Models (DDPMs).

In a DDPM, the forward process gradually corrupts a data sample x_0 into nearly pure Gaussian noise x_T over T discrete steps using a fixed Markovian noise schedule. The reverse process is parameterized by a neural network, typically a U-Net, which learns to predict the added noise at each step and thereby approximate the true reverse diffusion kernel. Sampling from a DDPM consists of iteratively denoising a Gaussian sample step by step until an image emerges. While DDPMs produce high-quality samples, they usually require hundreds or thousands of reverse steps to generate one image, making inference slow.

Song et al. [33] proposed Denoising Diffusion Implicit Models (DDIMs) as a non-Markovian reformulation of the reverse process. DDIMs keep the same learned score network as a DDPM but use a different deterministic sampling procedure that integrates the corresponding reverse-time stochastic differential equation. This allows the model to generate samples in far fewer steps (dozens instead of thousands) without retraining, at the cost of introducing a small discrepancy from the exact DDPM distribution. DDIM sampling can also be interpreted as a form of ancestral sampling with controllable stochasticity, which provides a trade-off between fidelity and diversity.

In practice, most modern latent diffusion frameworks are trained using the standard DDPM objective but can be sampled either with the original DDPM schedule for maximum diversity or with a DDIM schedule for faster image generation.

2.2.2 Latent Diffusion Models: High-Quality Image Generation in the Latent Space

Latent diffusion models (LDMs) were proposed by Rombach et al. in 2022 [13], as an improvement on standard diffusion models by performing the modeling on a latent space.

We can interpret traditional DMs as an equally weighted sequence of denoising autoencoders $\epsilon_\theta(x_t, t), t = 1, ..., T$, trained to predict x, a denoised variant of the noisy variant x_t. The corresponding objective is given by:

$$L_{DM} := \mathbb{E}_{\epsilon,\epsilon\sim N(0,1),t}[||\epsilon - \epsilon_0(x_t, t)||_2^2] \tag{2.15}$$

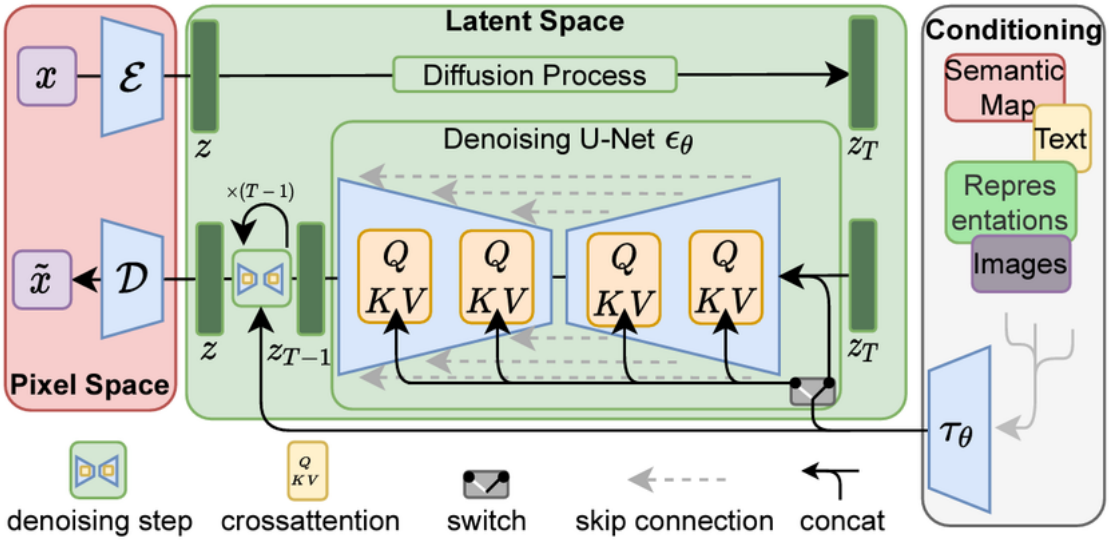


Figure 2.7: Schematic overview of the LDM architecture, from the original paper by Rombach et al. [13]. Input images x are first mapped from pixel space into a lower-dimensional latent representation z through an encoder $\mathcal{E}$. A diffusion process is then applied in the latent space, where a denoising U-Net ϵ_θ iteratively removes noise using cross-attention layers (Q, K, V) and skip connections. Conditioning information, such as semantic maps, text prompts, image features, or other representations, is incorporated into the denoising process through a conditioning network τ_θ. Finally, the denoised latent representation $\tilde{z}$ is mapped back to pixel space by a decoder $\mathcal{D}$, reconstructing the synthetic image.

Considering the perceptual compression model [34] that, given an image $x \in \mathbb{R}^{HxWx3}$ in RGB color space, uses an encoder $\mathcal{E}$ to encode x into a latent representation $z = \mathcal{E}(x)$ and a decoder $\mathcal{D}$ to reconstruct the image from z, outputting $\tilde{x} = \mathcal{D}(z)$ with $z \in \mathbb{R}^{hxwxc}$, a lower dimension space. The corresponding objective is given by:

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(\epsilon),\epsilon\sim N(0,1),t}[||\epsilon - \epsilon_0(z_t, t)||_2^2] \tag{2.16}$$

Additionally, LDMs are made more flexible as conditional image generators by integrating a cross-attention mechanism into their UNet backbone. This adaptation enables them to process and learn from various input modalities more effectively. To achieve this, a

domain-specific encoder, denoted τ_θ, is introduced. This encoder projects y to a intermediate representation $\tau_\theta(y) \sim \mathbb{R}^{Mxd_\tau}$, and then map it onto the intermediate layers of the UNet through a cross-attention layer, which we will now explain.

In the LDM scheme in Fig. 2.7, each block of the U-Net includes a *Cross-Attention (CA) layer* to inject information from the conditioning signal (such as an identity context in the context of generating face images) into the latent feature map. Unlike self-attention (for more details on self-attention, check Appendix A.3), where queries, keys and values are all derived from the same sequence, here the queries (Q) come from the current latent features of the U-Net, while the keys (K) and values (V) come from the conditioning embedding.

Let d_k be the dimensionality of the key and query vectors. The CA operation computes the similarity between each query from the latent feature map and the keys from the conditioning embedding, and uses these similarities to retrieve and blend the corresponding values, as in Equation 2.17, allowing each spatial location in the latent feature map to attend to the most relevant information in the conditioning context at each layer of the U-Net.

$$\text{CA}(Q_{\text{lat}}, K_{\text{cond}}, V_{\text{cond}}) = \text{softmax}\left(\frac{Q_{\text{lat}} K_{\text{cond}}^{\top}}{\sqrt{d_k}}\right) V_{\text{cond}} \tag{2.17}$$

This process is repeated in parallel across *multiple heads*, each learning different interaction patterns between latent features and conditioning tokens. The outputs from all heads are concatenated and linearly projected. The result of this linear projection is then fed into the next residual and convolutional layers of the U-Net.

Finally, a position-wise Feed-forward Network (FFN) (see Appendix A.1.4), typically consisting of two linear layers with a non-linear activation (such as ReLU) in between, refines each position's representation independently. This FFN adds expressivity to the model while preserving the conditioning information introduced by cross-attention.

2.3. Face Recognition (FR) systems

In the 1990s, the study of FR gained popularity due to the introduction of the Eigenface approach [35]. Holistic approaches dominated the FR community in the 1990s and 2000s, attempting to recognize human faces using one or two layer representations [36] [37] [38]

[39]. However, it relied on deriving low-dimensional representations based on distributional assumptions, resulting in these methods struggling to handle uncontrolled facial changes that deviated from these assumptions.

Then, in the early 2000s, FR using handcrafted local descriptors such as Gabor [40] and LBP [41] gained popularity, followed by local feature learning approaches in the early 2010s [42] [43] [44]. These methods improved FR accuracy of the Labeled Faces in the Wild (LFW) [45] benchmark to only about 95%, even after over a decade of work. Moreover, these systems often addressed only one type of variability in facial recognition, such as changes in lighting, pose, expression, or disguise. They lacked an integrated approach to handle all these challenges simultaneously, which is crucial in real-world, unconstrained scenarios. This led to both an unstable performance and frequent false alarms in practical applications.

In 2014, Deep Learning caused a major shift in the research of FR by employing deep CNNs in FR. This boosted the FR performances to an unprecedented level, due to advanced network architectures [46] [47] [48] [49], and discriminative learning methodologies such as DeepFace [50], FaceNet [51] and DeepID [52].

Typically, FR can be categorized as *face identification* (classify a face to a specific identity) and *face verification* (determine whether a pair of faces correspond to the same identity) [53]. Moreover, in terms of the testing protocol, FR systems can be evaluated under *closed-set* or *open-set* settings [54].

In closed-set testing, all identities in the test set were predefined during training, and so FR can be seen as a classification problem, assigning each face to a known identity, therefore features are expected to be separable. For open-set protocols, the testing identities are usually disjoint from the training set, making a classification approach impossible. In this case, face identification is similar to face verification between the face in the testing set and every other identity, and so open-set testing becomes a metric learning problem, with a main objective of learning discriminative large-margin features in order to effectively distinguish between known and unknown identities.

As shown in Fig. 2.8, closed-set FR at inference time is essentially a standard classification problem over a fixed set of existing identities, while open-set FR behaves like a metric-learning problem, since it is evaluated by comparing distances between embeddings, allowing identities not seen during training.

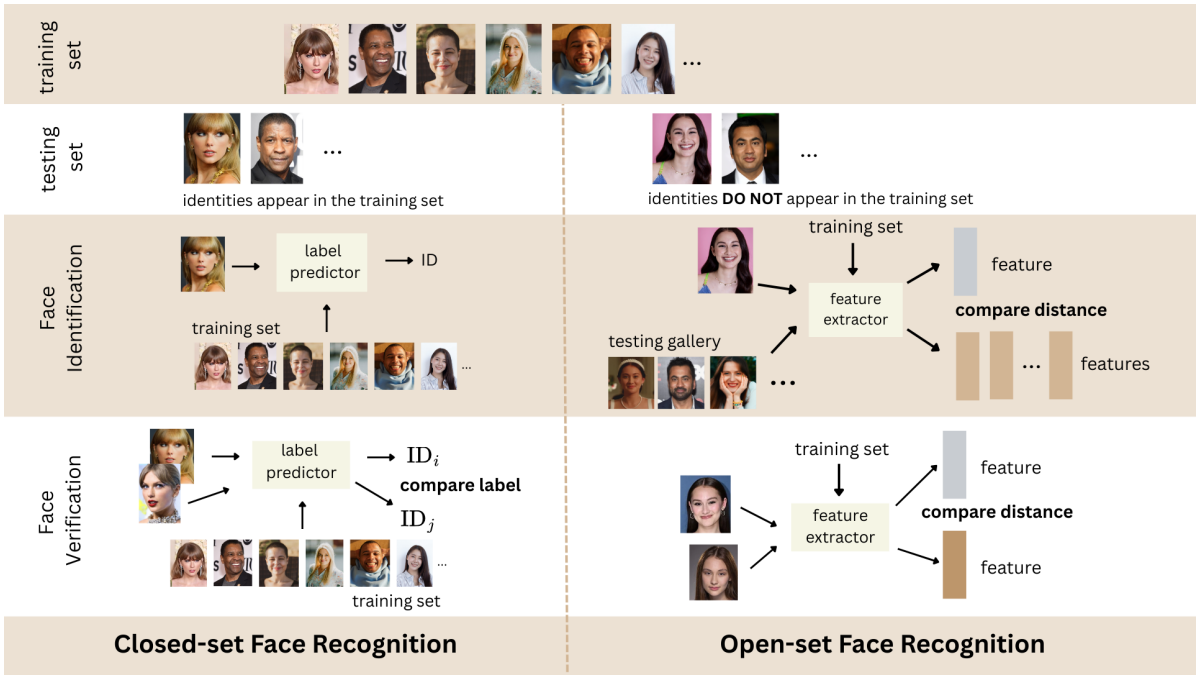


Figure 2.8: Face recognition systems: verification vs. identification (closed vs. open set).

During training, both open-set and closed-set deep FR models are typically trained as large classifiers: during training each identity in the dataset is treated as a separate class, and the network learns to assign the correct class to each input face (Fig. 2.9). The features from the last hidden layer, before the final classification layer, form a compact embedding for each face. Because the embeddings are normalized, they can be seen as points on the surface of a hypersphere (Fig. 2.10). This geomeFaces of the same person should produce embeddings that are close together, while faces of different people should be far apart.

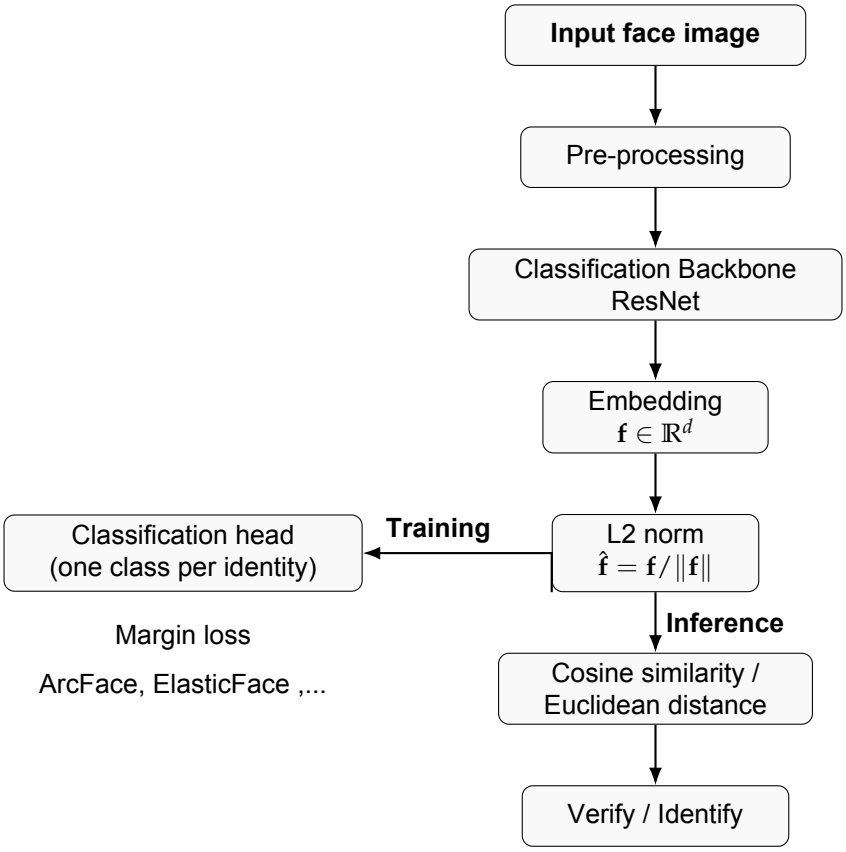


Figure 2.9: Schematic of the training and inference stages in a face-recognition pipeline. During training, normalized embeddings are passed to a classification head and optimized with a margin-based loss (e.g. ArcFace [17], ElasticFace [20]) to produce a discriminative embedding space. During inference, for open-set verification and identification tasks, the classification head is discarded and faces are compared by measuring distances between normalized embeddings. In closed-set tasks, however, the classification head is kept and a softmax decision is made over a fixed set of identities.

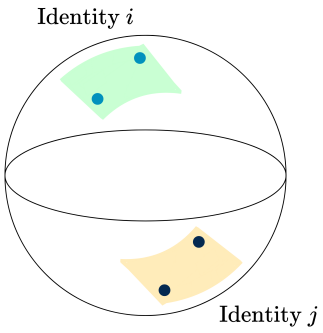


Figure 2.10: Illustration of the embedding space learned by a FR model. Each colour represents the cluster of embeddings for one identity on the surface of a hypersphere. Tight clusters indicate low intra-class variation, while large angular gaps between clusters indicate high inter-class separability.

During training, special loss functions are used to make each identity form a tight cluster and to increase the gap between clusters. Once this embedding space is learned, comparing two faces reduces to measuring the distance between their embeddings. We start

with exploring the standard softmax loss used in classification problems and then review domain-specific losses designed for face recognition, such as SphereFace [19], CosFace [18], ArcFace [17], and ElasticFace [20].

2.3.1 Loss functions in deep FR models

Let's suppose we have a simple image classification problem with C classes. Given a d-dimensional input $\mathbf{x} \in \mathbb{R}^d = \{x_1, ..., x_d\}$ (for example, for a 2×2 gray-scale image, $d = 4$), the model outputs a raw score z_{y_i} for each class $y_i = y_1, ..., y_C$, so the output is, in vector notation, $\mathbf{z} = \mathbf{W}\mathbf{x} + \mathbf{b} = \{z_{y_i}\}_{i=1,...,N}$, where $\mathbf{W} \in \mathbb{R}^{C \times d}$ is the weight matrix and $\mathbf{b} \in \mathbb{R}^C$ is the bias vector. y_i is a one-hot vector, i.e., for each image i, $y_i = 1$ for a correct classification and $y_i = 0$ otherwise (for example, for $C = 3$, $y_i \in \{(0,0,1),(1,0,0),(0,1,0)\}$).

The softmax function $\text{softmax}(\mathbf{z}) : \mathbb{R}^C \to [0,1]^C$ converts each score z_{y_i} into a probability p_{y_i} of belonging to class y_i, as described in Equation 2.18.

$$p_{y_i} = \mathbb{P}(y_i|\mathbf{x};\mathbf{W},\mathbf{b}) = \frac{\exp z_{y_i}}{\sum_{j=1}^{C} \exp z_{y_j}} \tag{2.18}$$

This ensures that $p_{y_i} \in [0,1]$ and $\sum_{i=1}^{C} p_{y_i} = 1$. Additionally, using $p_{y_i} \propto \exp z_{y_i}$ forces the conditional class probability to increase with increasing z_i and is monotonic, so $\text{argmax}[p_{y_i}] = \text{argmax}[z_i]$.

Assume now we have a batch of N inputs $\mathbf{X} \in \mathbb{R}^{d \times N}$. Now we can use $\mathbf{Z} = \mathbf{X}\mathbf{W} + \mathbf{b}$, with $\mathbf{W} \in \mathbb{R}^{N \times d \times C}$ and $\mathbf{b} \in \mathbb{R}^C$. Let $\hat{\mathbf{Y}} = \text{softmax}(\mathbf{Z})$.

We can now compare the estimated conditional probabilities $\mathbf{p} = \hat{\mathbf{y}}$ for a dataset with features $\mathbf{X}$ and labels $\mathbf{Y}$ one-hot encoded by checking how probable the actual classes are according to our model.

$$\mathbb{P}(\mathbf{Y}|\mathbf{X}) = \prod_{i=1}^{N} \mathbb{P}(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) \tag{2.19}$$

Our goal is to maximize the probability in Equation 2.19. However, since maximizing the product of terms is difficult, we use the negative logarithm and, equivalently, minimize the negative log-likelihood.

$$-\log \mathbb{P}(\mathbf{Y}|\mathbf{X}) = \sum_{i=1}^{N} -\log \mathbb{P}(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}) = \sum_{i=1}^{N} l(\mathbf{y}^{(i)}, \hat{\mathbf{y}}^{(i)}) \tag{2.20}$$

Where, for any pair of label and model prediction, the expected loss over classes is

$$l(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{j=1}^{C} y_j \log \hat{y}_j \tag{2.21}$$

$$= -\sum_{j=1}^{C} y_j \log \frac{\exp z_{y_i}}{\sum_{j=1}^{C} \exp z_{y_j}} \tag{2.22}$$

$$= \sum_{j=1}^{N} y_j \log \sum_{j=1}^{C} \exp z_{y_j} - \sum_{j=1}^{C} y_j z_{y_i} \tag{2.23}$$

$$= \log(\sum_{j=1}^{C} \exp z_{y_j})^{\sum_{j=1}^{C} y_j} - \sum_{j=1}^{C} y_j z_{y_i} \tag{2.24}$$

$$= \log(\sum_{j=1}^{C} \exp z_{y_j}) - \sum_{j=1}^{C} y_j z_{y_i} \tag{2.25}$$

Since y_i is one-hot encoded, the expression for $l(\mathbf{y}, \hat{\mathbf{y}})$ in Equation 2.22 collapses to a single term

$$l(\mathbf{y}, \hat{\mathbf{y}}) = -\log \frac{\exp z_{y_i}}{\sum_{j=1}^{C} \exp z_{y_j}} \tag{2.26}$$

Therefore, the average softmax loss the minibatch of size N is given by

$$L_{Softmax} = \frac{1}{N} \sum_{i=1}^{N} l(\mathbf{y}^{(i)}, \hat{\mathbf{y}}^{(i)}) \tag{2.27}$$

$$= \frac{1}{N} \sum_{i=1}^{N} -\log \frac{\exp z_{y_i}}{\sum_{j=1}^{C} \exp z_{y_j}} \tag{2.28}$$

$$= -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp (W_{y_i}^T x_i + b_{y_i})}{\sum_{j=1}^{C} \exp (W_j^T x_i + b_j)} \tag{2.29}$$

The fact that sofmax loss is easily interpretable and optimizable (gradient descent), and that it encourages clear decision boundaries, makes it very useful in classification tasks. However, it is not optimal for FR tasks, since it lacks explicit feature separability. Geometrically, softmax just separates classes by decision boundaries without enforcing a margin. Imagine embedding vectors normalized onto a hypersphere. Softmax encourages samples to be closer to their class center, but does not control how far apart the classes are. This can lead to overlaps when classes are close.

2.3.1.1 Angular softmax (SphereFace)

Angular Softmax (A-Softmax), introduced by Liu et. al. [19] in 2018, is a loss function designed to enable deep CNNs to learn angularly discriminative features (SphereFace) for FR. The authors introduce a multiplicative angular margin in the loss function, controlled by a parameter m, that penalizes angles between features and their class weights, forcing intra-class features to cluster more tightly and inter-class features to separate by a larger angular distance.

Considering the softmax loss in Equation 2.29, the authors start by normalizing the weight vectors ($||W_j|| = 1$) and removing biases ($b_j = 0$).T« Then, the authors proceed to transform the logits $z_j = W_j^T x_i$ (since $b_j = 0$) into angular terms, with θ_j being the angle between x_j and W_j, as described in Equation 2.30.

$$W_j^T x_i = ||x_i|| \cos\theta_j \tag{2.30}$$

Equation 2.31 shows the modified softmax loss function after these transformations.

$$L_{modified} = -\frac{1}{N}\sum_{i=1}^{N} \log\left(\frac{\exp\left(||x_i|| \cos\theta_{y_i,i}\right)}{\sum_{j=1}^{C} \exp\left(||x_i|| \cos\theta_{j,i}\right)}\right) \tag{2.31}$$

The A-Softmax loss introduces the multiplicative angular margin m mentioned previously, as can be seen in Equation 2.32. For the correct class y_i, $\cos\theta_{y_i}$ is replaced by $\cos m \cdot \theta_{y_i}$, tightning the angular constraint. In this Equation, $\phi(\theta_{y_i,i}) = (-1)^k \cos m \cdot \theta_{y_i} - 2k$ for $\theta_{y_i,i} \in \left[\frac{k\pi}{m}, \frac{(k+1)\pi}{m}\right]$ and $k \in [0, m-1]$.

$$L_{A-Softmax} = -\frac{1}{N}\sum_{i=1}^{N} \log\left(\frac{\exp\left(||x_i||\phi(\cos\theta_{y_i,i})\right)}{\exp\left(||x_i||\phi(\cos\theta_{y_i,i})\right) + \sum_{j=1,j\neq y_i}^{C} \exp\left(||x_i|| \cos\theta_{j,i}\right)}\right) \tag{2.32}$$

As for its geometric interpretation, for binary classification, the decision boundary for class 1 becomes $\cos m \cdot \theta_1 = \cos\theta_2$, enforcing an angular margin of $\frac{m-1}{m+1}\theta_{1,2}$, where $\theta_{1,2}$ is the angle between W_1 and W_2.

In summary, the idea behind A-Softmax is to enforce an angular margin between classes. Instead of just maximizing cosine similarity between a sample and its correct class (identity), A-Softmax forces the angle between classes to be wider. On the hypersphere, that

means samples of one class must be closer together, leaving a greater separation between classes.

Experimentally, SphereFace outperformed softmax, contrastive, triplet, and center losses [19].

2.3.1.2 Large Margin Cosine Loss (CosFace)

Wang et. al. [18] propose, in 2018, a novel loss function, Large Margin Cosine Loss (LMCL), to enhance feature discrimination in deep face recognition. The core idea is to reformulate the traditional softmax loss into a cosine-based loss by incorporating a fixed margin in the angular space.

Similar to SphereFace, CosFace normalizes the weight vectors ($||W_j|| = 1$) and removes biases ($b_j = 0$), transforming logits into angular terms (Equation 2.33):

$$W_j^T x_i = \cos\theta_j, \tag{2.33}$$

, where θ_j is the angle between x_i and W_j. Additionally, CosFace normalizes the feature vectors as well ($||x_i|| = s$), projecting both weights and features onto a hypersphere of radius s. Since this removes the variations in radial directions, the model learns features that are separable in the angular space.

The modified softmax loss with normalization (NSL) becomes:

$$L_{\text{ns}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left(s\cos\theta_{y_i,i}\right)}{\sum_{j=1}^{C}\exp\left(s\cos\theta_{j,i}\right)}\right) \tag{2.34}$$

However, NSL only emphasizes correct classification. To further enhance the discriminative power of this model, the authors introduced a cosine margin m to the classification boundary by replacing $\cos\theta_{y_i,i}$ wit h$\cos\theta_{y_i,i} - m$. The corresponding Large Margin Cosine Loss (LMCL) is given by Equation 2.35.

$$L_{\text{lmcl}} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left(s(\cos\theta_{y_i,i} - m)\right)}{\exp\left(s(\cos\theta_{y_i,i} - m)\right) + \sum_{j\neq y_i}\exp\left(s\cos\theta_{j,i}\right)}\right). \tag{2.35}$$

For binary classification, the decision boundary for class 1 becomes $\cos\theta_1 - m = \cos\theta_2$, enforcing a fixed cosine margin m between the classes. This differs from the SphereFace

boundary- $\cos m \cdot \theta_1 = cos\theta_2$, which creates a non-monotonic θ dependent angular margin. LMCL's additive margin ensures uniform separation across all classes, avoiding optimization difficulties inherent in angular margin formulations.

Geometrically, CosFace compresses features from the same identity into tighter clusters on the hypersphere, while pushing features from different identities apart by a margin proportional to m (impostors need a significantly higher similarity to cross the boundary).

CosFace outperforms SphereFace, achieving a 99.73% accuracy in LFW, while SphereFace achieved an accuracy of 99.42%. Empirically, the margin $m = 0.35$ is optimal, and values of m above decrease the model's performance due to excessively tight constraints.

2.3.1.3 Additive Angular Margin Loss (ArcFace)

Additive Angular Margin Loss, introduced in 2022 by Deng et. al., addresses limitations of traditional softmax loss and the previously mentioned margin-based methods, SphereFace and CosFace.

The core idea behind ArcFace is to modify the softmax loss function by adding a fixed angular margin m to the angle between the feature vector and the target class weight vector.

From the softmax loss expression in Equation 2.29, after removing biases $b_j = 0$ and normalizing the weights $||W_j|| = 1$ using L_2 normalization, the authors fix the embedding features $||x_i||$ using L_2 normalization and re-scaling them to s, they distribute the learned embedding features on a hypersphere with a radius of s (Equation 2.36).

$$L_{ArcFace} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left(s\cos(\theta_{y_i,i})\right)}{\exp\left(s\cos(\theta_{y_i,i})\right) + \sum_{j=1,y\neq y_i}^{C}\exp\left(s\cos\theta_j\right)}\right) \tag{2.36}$$

After the embedding features are distributed around each feature center on the hypersphere, they add an additive angular margin penalty m between x_i and W_{y_i} to enhance inter-class discrepancy and intra-class compactness.

$$L_{ArcFace} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left(s\cos(\theta_{y_i,i} + m)\right)}{\exp\left(s\cos(\theta_{y_i,i} + m)\right) + \sum_{j=1,y\neq y_i}^{C}\exp\left(s\cos\theta_j\right)}\right) \tag{2.37}$$

On the hypersphere, this means that each identity class is not only clustered compactly, but also separated from other classes by a fixed angular gap. Compared to LMCL, which

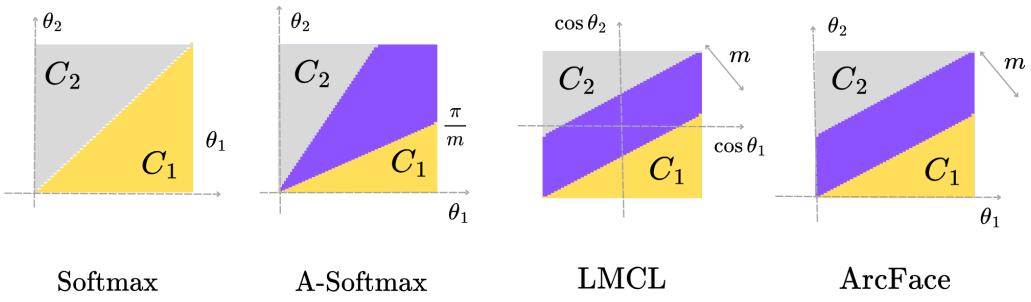


Figure 2.11: Decision margins of different losses for binary classification: Softmax, A-Softmax, LMCL, and ArcFace. The decision margins are represented in purple, and grow progressively stricter from Softmax $\rightarrow$ A-Softmax $\rightarrow$ LMCL $\rightarrow$ ArcFace, illustrating the progression from no margin (Softmax) to increasingly strict angular constraints, which improve intra-class compactness and inter-class separability.

shifts the cosine similarity threshold, ArcFace ensures that decision boundaries correspond to a clear geometric interpretation: samples must lie within a tighter angular region around their class center, leading to more discriminative embeddings.

ArcFace outperformed CosFace and SphereFace in face verification accuracy, achieving 99.53% accuracy in LFW, with optimal empirical results at $m = 0.5$.

2.3.1.4 ElasticFace

Models such as SphereFace [19], CosFace [18], ArcFace [17] use advanced versions of L-Softmax loss that add fixed penalty margins to the decision boundary. However, fixed penalty margins, such as ArcFace's angular margin, assume that all identities are equally distributed in geodesic space, which is usually not the case in real datasets. Additionally, in real datasets, some classes have higher intra-class variation than others, making a fixed margin suboptimal.

Boutros et. al. introduce ElasticFace [20] for this reason, that uses a margin penalty that is randomly sampled from a Gaussian distribution during training (see Equation 2.38, where μ is the mean margin- 0.5 for ArcFace based models-, and σ is the standard deviation, controlling the randomness of the margin).

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp -\frac{(x-\mu)^2}{2\sigma^2} \tag{2.38}$$

Let $E(m, \sigma)$ be a function that returns a random value from a normal distribution with mean m and standard deviation σ.

1. ElasticFace-Arc and ElasticFace-Cos

   We have seen the definitions for both ArcFace (Equation 2.37) and CosFace losses (Equation 2.35). Replacing m with the normal function $E(m, \sigma)$, we get the ElasticFace-Arc (Equation 2.39) and ElasticFace-Cos losses (Equation 2.40).

$$L_{EArc} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left(s\cos(\theta_{y_i,i} + E(m,\sigma))\right)}{\exp\left(s\cos(\theta_{y_i,i} + E(m,\sigma))\right) + \sum_{j=1,y\neq y_i}^{C}\exp\left(s\cos\theta_j\right)}\right) \tag{2.39}$$

$$L_{ECos} = -\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{\exp\left(s\cos(\theta_{y_i,i} - E(m,\sigma))\right)}{\exp\left(s\cos(\theta_{y_i,i} - E(m,\sigma))\right) + \sum_{j=1,y\neq y_i}^{C}\exp\left(s\cos\theta_j\right)}\right) \tag{2.40}$$

   ElasticFace-Arc has as decision boundary $\cos(\theta_{y_i} + E(m,\sigma)) - \cos\theta_j = 0$ and ElasticFace-Cos's is $\cos\theta_{y_i} - \cos\theta_j - E(m,\sigma) = 0$

2. ElasticFace+

   The authors propose an extension to ElasticFace, which they name ElasticFace+. ElasticFace+ uses the observed intra-class variation in each training step to assign a value for the margin to each sample based on how close it is to the class center. To achieve this, they they sort the output of the Gaussian distribution function in a descending order based on the value of $\cos\theta_{y_i}$- a sample with a small $\cos\theta_{y_i}$ is consequentially pushed with a large value of $E(m,\sigma)$, and vice-versa.

2.4. Synthetic data for Face Recognition

According to Boutros et al. [55], there are four use cases for synthetic face datasets in terms of their interaction with FR systems: training FR models, evaluating their performance, conducting attacks on FR systems, and enhancing privacy for FR users. Below, we talk about the two most relevant ones in the contexts of this thesis: *training* and *evaluating* FR systems.

- **Training Face Recognition Models**

Modern Face Recognition systems typically rely on Deep Learning models that are trained either to directly learn identity-discriminative features (using methods such as triplet loss [51]) or to classify the training data's identity classes (e.g. ArcFace [17], ElasticFace [20]) using the embeddings from the penultimate layer of the network to extract the identity-discriminant representations. These two approaches both need to be trained using data that presents both high inter-class and intra-class diversity, in order to reflect real-world variations.

Synthetic data can be especially useful to train such FR models when tailored according to the training data requirements. If the training of the model is done in one of the approaches mentioned above, the synthetic data should include a large number of distinct identities along with multiple samples per identity [56] [57].

When the training process is done using partially authentic data, this implies insufficient intra-class variation, so synthetic data must be used as an augmentation technique, with multiple samples per authentic identity [56] [58].

Finally, in unsupervised training scenarios, where precise identity grouping is not as concerning, a set of faces of random identities is sufficient. Additionally, this data has also proven effective in training-aware quantization of models using full precision parameters [59].

- **Evaluation of Face Recognition Systems**

  The evaluation of FR algorithms requires a large set of genuine (same-identity) and impostor (different-identity) face image pairs that accurately portray real-world operational scenarios. The requirement of synthetic data for evaluating FR systems is largely due to the fact that, as FR systems achieve progressively lower false match rates (FMR) and false non-match rates (FNMR), the number of pairs to make the evaluation results statistically significant increases accordingly.

  Another situation where synthetic face data would be a better alternative to real data is that some authorities only possess a single image per identity, making it impossible to have genuine pairs to evaluate FR algorithms. These two situations call for generation of synthetic data belonging to a certain authentic identity, but containing realistic variations.

  Another scenario for the use of synthetic face data would be for operations requiring a very low FMR, therefore needing a very large number of impostor pair in order to

evaluate the FMR with statistical significance. In this case, random synthetic faces with random identities can be used to generate the needed impostor pairs.

What type of synthetic data is needed for each use case and what qualities make it good? Additionally, the types of synthetic data that are needed for training and evaluating FR systems are described below [55].

- **Single Face Images of Random Identities**

  For unsupervised training or for evaluating the false match rate (FMR), you need individual face images of random identities, since the generated images must appear as realistic as authentic face data. This is so that, when used in a face recognition model, they bring out comparable activation responses. Moreover, the distribution of similarity scores (especially for impostor comparisons) should mirror that of authentic datasets.

- **Multiple Face Images per Random Identity**

  Supervised training of FR models must use multiple images per identity in order to capture both inter-class differences and intra-class variations (such as changes in pose, expression, and lighting). Capturing these variations ensures that models can are robust against genuine and impostor pairings. The synthetic samples should be statistically comparable to real data, ensuring that any evaluation metric reflects the targeted real-world variability.

- **Multiple Faces of An Existing Identity**

  In cases where an authentic dataset has an insufficient variation for a given identity (intra-class variation), synthetic augmentation is essential to ensure the efficiency of the training and evaluation of FR systems. Therefore, the generated images should introduce the realistic variations, such as facial expression, angle, or illumination, that the original data lacks, helping models become robust to intra-class variations and, consequently reducing false non-match rates in practical applications.

2.4.1 Existing approaches for generating synthetic face data

GAN-based approaches With the large success of GANs in image generation, several methods were proposed to generate identity-preserving face images, namely FF-GAN [60], FaceID-GAN [61], DiscoFaceGAN [62]). However, the poor intra-class variability

and the domain gap between the synthetic and real face datasets motivated Qiu et al. [56], in 2021, to develop SynFace, incorporating identity and domain mix-ups into the generation process. In June 2022, Boutros et al. [57] proposed SFace, a GAN-generated class-labeled synthetic dataset for training supervised FR models. In October 2022, Bae et al. [58] propose rendering digital faces using a computer graphics pipeline, introducing a large scale synthetic dataset for FR, DigiFace-1M. FR models trained on this dataset outperformed those trained on SynFace in terms of accuracy. In November 2022, Boutros et al. [59] proposed an unlabeled dataset for unsupervised FR models, USynthFace.

Diffusion-based approaches In 2023, Kim et al. [63] proposed the diffusion based Dual Condition Face Generator (DCFace) that combines subject appearance (ID) and external factor (style) conditions, as a way to control both the inter-class and the intra-class variations. Also in 2023, Boutros et al. [12] proposed an approach based on conditional latent diffusion models for synthetic identity generation with realistic identity variations, IDiff-Face, achieving a 98.00% accuracy on the LFW benchmark and an authentic-based face recognition with 99.82% accuracy. In 2024, Yeung et al. proposed VariFace [64], a two-stage diffusion-based pipeline to create fair and diverse synthetic face datasets to train face recognition models, which achieved a 98.87% accuracy on the LFW benchmark. TCDiff, introduced in September 2024 by Biesseck et al. [65], is a diffusion model that incorporates both 2D and 3D facial constraints to stylize synthetic faces. By blending an identity image with a style image—applying these dual constraints, TCDiff enhances identity preservation while maintaining style variation. In 2025, Mi et al. introduced MorphFace [66], a diffusion framework that combines identity conditioning with 3DMM-derived style embeddings, such as pose and expression. By sampling realistic distributions of these styles and blending them with identity information, MorphFace achieved a better balance between identity preservation and style diversity. In August 2025, Caldeira et al. proposed NegFaceDiff [67], a novel negative conditioning approach during diffusion sampling. Essentially, the model is guided not only by the target identity but also away from unwanted identity features, helping it avoid blending attributes that could cause confusion between identities.

Hybrid approaches Hybrid approaches have also emerged. In 2025, GANDiff FR [68] was introduced, combining StyleGAN3 [69] for identity control with diffusion layers for fine-grained manipulation of pose, illumination, and expression. This hybrid setup enabled

precise bias attribution and fairness analysis while reducing group disparity metrics.

Other approaches In 2024, Wu et. al [70] proposed Vec2Face, an holistic model that uses a sampled vector as input and can generate and control face images and their features. In 2025, Wu et al. extended their earlier work with Vec2Face+ [71], which further improved identity consistency and separability through vector-based sampling and refined attribute control. Their synthetic datasets not only matched but in several cases surpassed real-data benchmarks such as CASIA-WebFace. HyperFace [72], introduced in 2024, takes a different approach to building synthetic face datasets. Instead of randomly sampling new identities, it works inside a pretrained FR embedding space (a hypersphere) and deliberately places synthetic identities so that they are spread out and well separated. After this placement step, a generator turns the embeddings into images. By maximizing the distance between identities, HyperFace produces synthetic datasets with stronger inter-class variation and better separability. Face-recognition models trained on HyperFace data reach performance close to, or even above, previous synthetic approaches.

2.4.2 Limitations

Early synthetic face generation approaches based on GANs achieved a notable degree of realism but were notoriously difficult to train, due to instability and mode collapse limiting both inter-class diversity (distinct identities) and intra-class diversity (variations per identity) [73] [74]. These issues can harm face recognition performance, as models trained on such data tend to overfit to a limited range of identities. Although better architectures and loss functions help, the basic training process of adversarial learning is still difficult.

Diffusion models have become the main approach for generating synthetic faces in face recognition. They improve label consistency, offer more control over identity variation, and reduce the accuracy gap with real-data training. However, there are still still some important drawbacks. The main one is speed: generating large, balanced datasets requires many denoising steps per image, making the process slow and computationally expensive. Even with acceleration methods like distillation, diffusion remains much slower than GANs [68].

Another limitation is identity separability and diversity. Diffusion-based datasets often struggle to balance variety within identities and clear differences between identities. While there are methods that improve this trade-off [63] [12], they still do not fully match the

quality of real data. Newer approaches like NegFaceDiff are emerging to address this by improving identity consistency and separation [67].

Moreover, attribute control is imperfect. Even with conditioning, generated faces do not always match the specified attributes, such as ethnicity or gender. This lack of fidelity makes it harder to rely on synthetic data for tasks that require balanced demographic representation [75].

2.5. IDiff-Face: Latent Diffusion for Identity-Conditioned Face Synthesis

In this work, we build upon the IDiff-Face model [12], which will serve as the foundation for all of our experiments and proposed extensions. IDiff-Face is a synthetic face data generation approach based on conditional LDMs, designed in order to address the trade-off between intra-class diversity (variations within the same identity) and cross identity discrimination (distinctiveness between identities) in synthetic face datasets. This model operates in the latent space of a pre-trained Vector-Quantized Generative Adversarial Network (VQGAN) autoencoder [76] (for more details, read Appendix A.5). The LDM is trained to iteratively denoise latent representations over T timesteps, guided by identity conditions.

Identity contexts are derived from feature embeddings of a pre-trained face recognition model (such as ElasticFace [20]) and are posteriously injected into the denoising U-Net [31] through cross-attention layers (for more details, read Section 2.2.2), conditioning the generation process on identity-specific information, and therefore ensuring that generated images maintain the identity characteristics of the input context. The training is done to minimize the Mean Squared Error (MSE) between the predicted noise by the U-Net and actual noise, with identity context integrated via cross-attention.

The authors also introduce Contextual Partial Dropout (CPD) as a way prevent overfitting to the identity context as well as enhance intra-class diversity. It works by, during training, randomly dropping components of the identity embedding with a probability p (usually 25% or 50%). This forces the model to generate diverse variations (different poses, expressions, lighting) while maintaining identity coherence. A lower CPD probability p enhances identity discrimination, while higher values for p enhance intra-class diversity.

2.5.1 Sampling

Two approaches are used to generate synthetic identities: *uniform sampling* and *two-stage generation*.

In *uniform sampling*, identity embeddings are sampled uniformly from a hypersphere (L2-normalized Gaussian vectors) to mimic the distribution of real face embeddings. Fixing the embedding and varying the noise seed generates multiple images of the same synthetic identity.

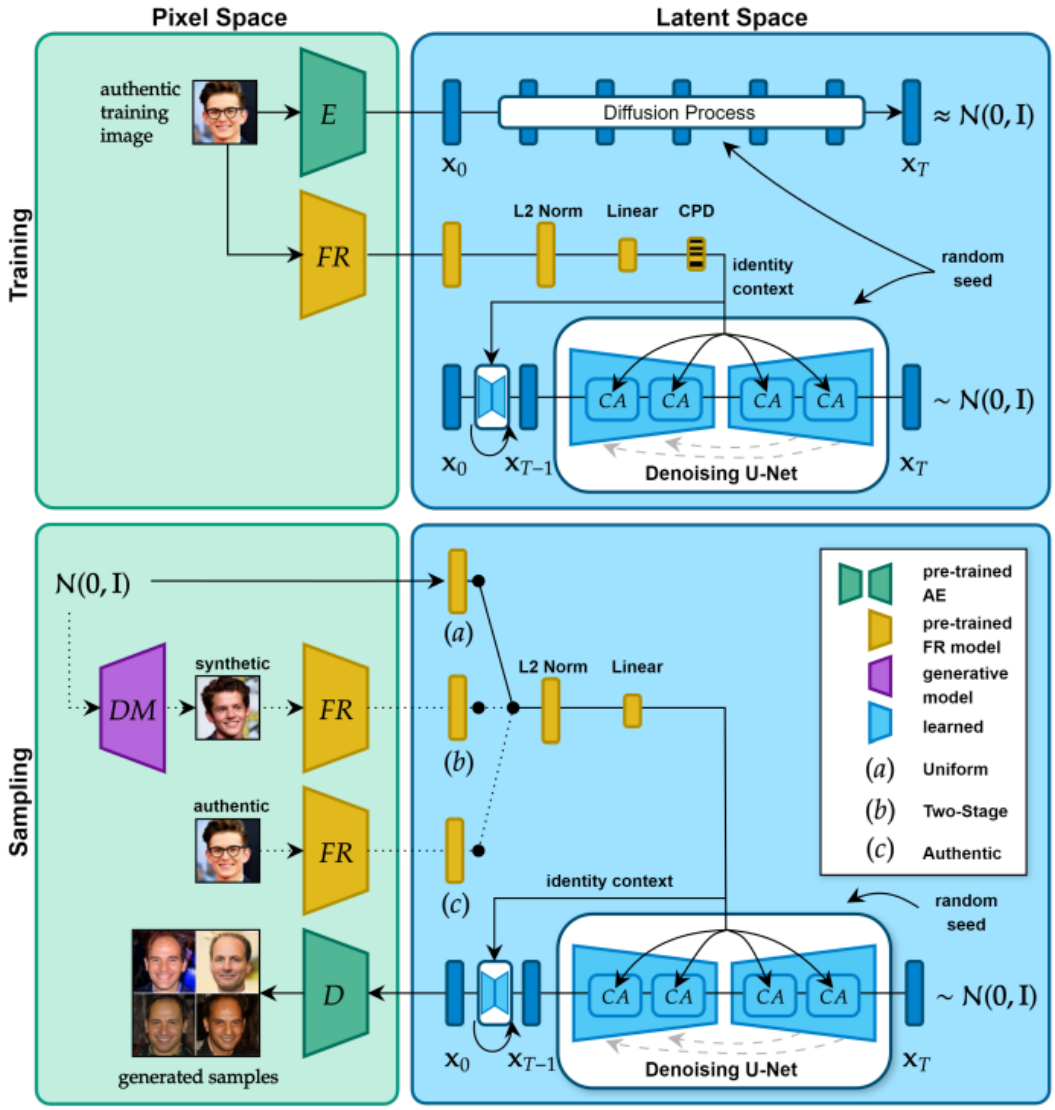


Figure 2.12: Overview of the proposed IDiff-Face. From the original paper. The model builds on a latent diffusion backbone, where noisy latent representations are progressively denoised by a U-Net architecture. Identity embeddings are injected into the denoising process through cross-attention, guiding the model to generate synthetic faces that preserve identity and exhibit realistic intra-class diversity. [12]

In *two-stage generation*, an unconditional LDM first generates a base synthetic image, and then the identity embedding of this image is extracted and used to condition IDiff-Face, producing variations of the same identity.

2.5.2 Identity conditioning in IDiff-Face

Instead of relying solely on random noise, IDiff-Face introduces *contextual embeddings* (identity representations), which are extracted using a pretrained FR model. The conditioning embeddings are injected into the U-Net backbone of the diffusion model through cross-attention layers (see Appendix , as follows:

- The input image (latent representation) produces queries (Q).
- The conditioning embeddings are projected into key (K) and value (V) vectors.
- The U-Net attends to these K and V vectors at each denoising step, integrating identity and demographic information into the generative process.

This cross-attention mechanism make sure that the generated samples stay consistent with the target identity (identity preservation) while still capturing realistic intra-class variability (different poses, expressions, lighting).

2.5.3 Advantages of IDiff-Face over GAN-based approaches

GANs often require explicit attribute control, such as pose and lighting. IDiff-Face, however, naturally generates realistic variations through the stochastic diffusion process and CPD, operating efficiently in the latent space, therefore enabling high-resolution generation with reduced computational costs.

In terms of performance, FR models trained on the IDiff-Face synthetic dataset achieve 98.00% verification accuracy on LFW [77], bridging the gap between synthetic and authentic data performance. The generated images exhibit natural variations in pose, expression, and accessories, while maintaining strong separability between identities (EER of 0.003 with two-stage CPD of 0%). Overall, IDiff-Face outperforms GAN-based methods like SynFace [56] and SFace [57], as well as rendering-based approaches such as DigiFace-1M [58] across multiple datasets (LFW, AgeDB-30 [78], CFP-FP [79]).

3. Methodology

In Section 3.1 we outline our objectives and provide an overview of the proposed pipeline and its evaluation. Then, in Section 3.3, we present the authentic face datasets used for training and evaluating our models. Next, in Section 3.4, we explain how we align our face images to the ArcFace geometry; in Section 3.5, we explain how we extract the ElasticFace embeddings from our aligned images. In Section 3.6 we explain how we train the ethnicity-classifier of face embeddings; in Section 3.7, we dive into the generation of random synthetic uniform identity contexts. Next, in Section 3.8, we present the single-model IDiff-Face approach, which will serve as a synthetic baseline for the proposed ethnicity-conditioned IDiff-Face pipeline, also presented in this chapter; Section 3.9 provides the training details of the IDiff-Face models. Finally, in Section 3.10, we detail the evaluation protocol and metrics used to assess photorealistic quality, identity separability, utility for FR training, and ethnicity consistency.

3.1. Overview and Goals

This project aims to propose a diffusion model based approach that is able generate a synthetic face dataset that can be used as an alternative to real face data in the training of FR models.

We begin by evaluating synthetic data generated using IDiff-Face [12]. Although more recent approaches have been proposed, we adopt IDiff-Face as our baseline approach due to its reproducibility, availability, and relevance as a widely recognized benchmark.

Building on this baseline, we extend the model by incorporating implicit ethnicity conditioning. Specifically, we train ethnicity-specific IDiff-Face models to generate face data with improved intra-class diversity, ethnicity consistency, and inter-class separability.

Due to computational limitations (see Section 3.2), we cannot provide a synthetic face dataset large enough to train FR models: we aim simply to establish a reproducible pipeline for synthetic face generation and to assess whether our samples are useful for face recognition.

Therefore, we will generate a synthetic dataset using our ethnicity-specific IDiff-Face approach and evaluate our samples in terms of: (i) *photorealistic quality*; (ii) *identity separability*; (iii) *utility for small-scale training of FR models*; (iv) *ethnicity consistency*.

3.2. Computational limitations

Due to limitations in available computational resources, it was not feasible to generate a synthetic dataset on the same scale as the face datasets in the literature. Instead, we opted to generate a smaller but representative subset of synthetic samples per ethnicity group. This strategy allowed us to conduct a controlled evaluation of the proposed approach while ensuring that experiments remained computationally efficient.

3.3. Datasets for training and evaluation

This work uses two authentic face datasets with complementary characteristics, enabling both the training and evaluation of the proposed models.

Racial Faces in-the-Wild (RFW), introduced by Wang et al. in 2019 [80], is a benchmark dataset made up by images organized into four testing subsets: African, Asian, Caucasian, and Indian. It is widely adopted in the literature for evaluating racial bias in face recognition systems and will be used as an evaluation set, namely for photorealism metrics and FR performance evaluation of our approach.

BUPT-BalancedFace, introduced by Wang and Deng in 2020 [81], was specifically designed to mitigate racial bias. It contains an identical number of identities and images for each of the four ethnicity groups (African, Asian, Caucasian, and Indian), making it particularly well suited for training ethnicity conditioned models such as the one proposed in this work (Our Approach).

By combining these two datasets, we ensure a balanced training stage (BUPT-BalancedFace) together with a realistic evaluation scenario (RFW), thereby strengthening the reliability and fairness of the experimental results.

3.3.1 Racial Faces in-the-Wild (RFW)

When evaluating the photorealistic quality of the samples, we use RFW as the ground-truth dataset when computing photorealistic metrics such as SSIM, MS-SSIM, FID, and LPIPS (see Section 3.10.2). Additionally, we used this dataset to evaluate the FR models we trained on our synthetic data, since it allowed us to evaluate the performance of the models on the different ethnicity groups (see Section 3.10.4).

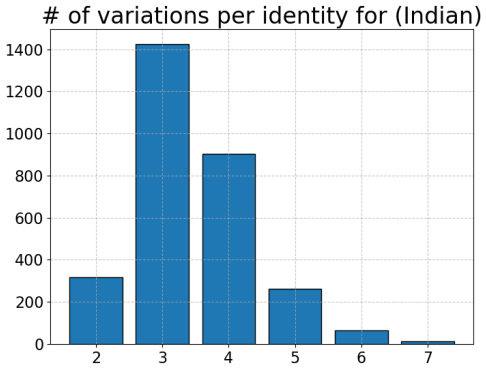
From Table 3.1, we can see that the RFW dataset is well balanced across the four ethnicity groups, with each containing approximately three thousand identities and around ten

Ethnicity	# of identities	# of images
Caucasian	2959	10196
Asian	2492	9688
Indian	2984	10308
African	2995	10415
Total	11430	40607

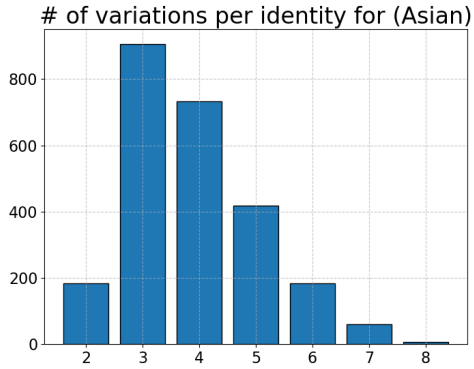
Table 3.1: Composition of the Racial Faces in-the-Wild (RFW) dataset, detailing the total number of identities and images for each ethnicity.

thousand images. The differences in the number of identities and total number of images across ethnicities are relatively small, which makes the dataset particularly suitable for evaluating fairness in FR systems.

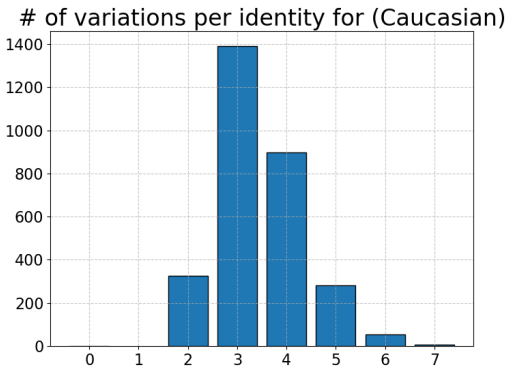
Note that, throughout this work, when we talk about “variations per identitity”, we are referring to the different images available for each identity (captured under different poses, lighting conditions, expressions, etc.).



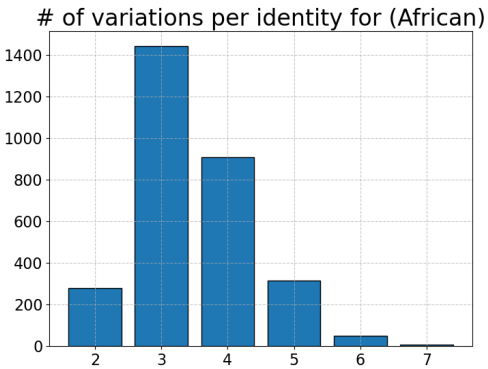
(a) Indian.



(b) Asian.



(c) Caucasian.



(d) African.

Figure 3.1: RFW: Distribution of the number of variations by ethnicity. The x-axis corresponds to the number of variations, while the y-axis represents the number of identities that have that number of variations.

As we can see in Fig. 3.1, the minimum number of variations per identity was 2 for every

ethnicity and the maximum was 8 for Asian images and 7 for the remaining ethnicities. The mode equals 3 in every ethnicity subset. Additionally, for the ethnicity groups Indian, Caucasian, and African the distribution of variations was more concentrated around the mode, indicating that most identities had a similar number of image variations. In contrast, the Asian group showed a slightly wider spread, with a relatively larger number of identities having more than 3 variations.

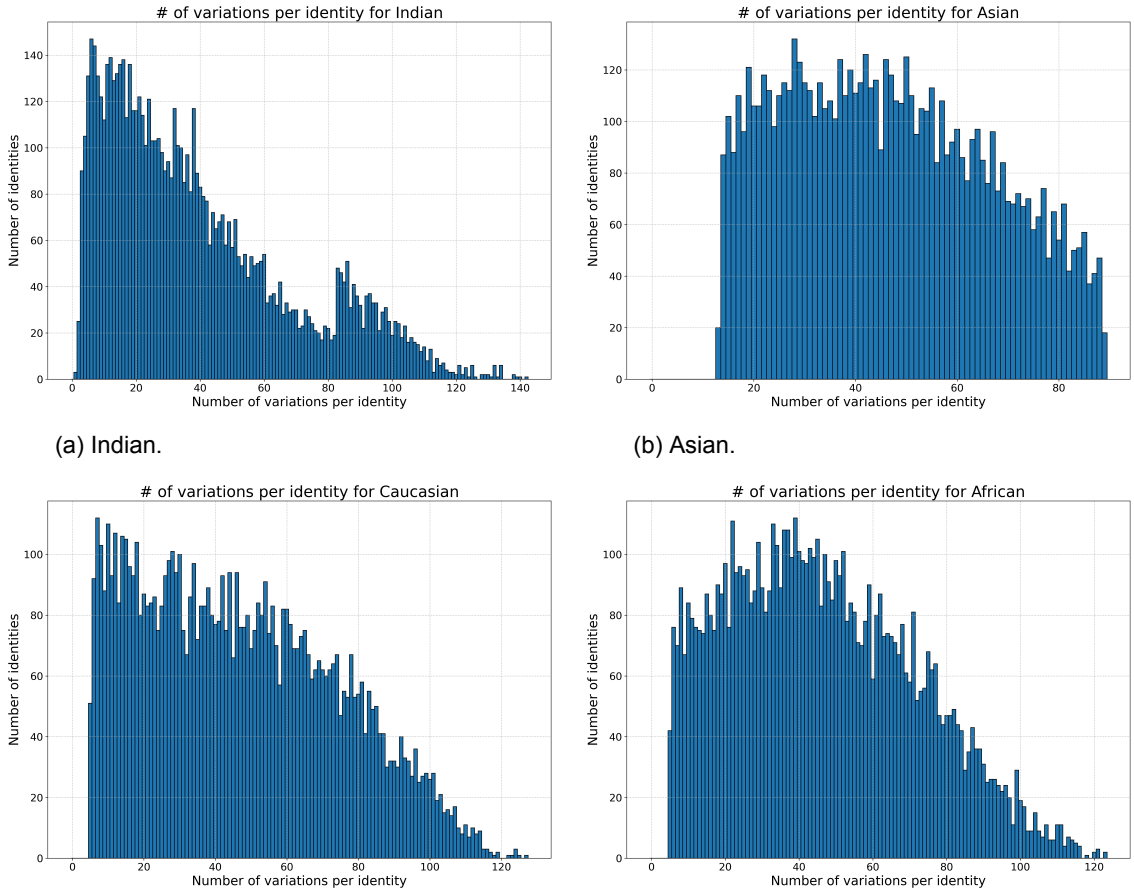
3.3.2 BUPT-BalancedFace

Throughout this work, to train our models, we use the BUPT-BalancedFace dataset, introduced by Wang et al. [81] in 2020, which is divided in four different ethnicities, African, Asian, Caucasian and Indian, each containing 7000 identities. The dataset's composition is shown in Table 3.2.

Ethnicity	# of identities	# of images
Caucasian	7000	326484
Asian	7000	325475
Indian	6997	275095
African	7000	324376
Total	27997	1251430

Table 3.2: Composition of the BUPT-BalancedFace dataset, showing the distribution of identities and images across the four ethnicity groups.

We also draw histograms (Figure 3.2) to better understand the distribution of the number of image variations available per identity in each ethnic group. These plots provide a more detailed view of how balanced the dataset is, beyond the raw counts of identities and images, allowing us to infer on whether certain groups contain many identities with only a small number of images, or if the data is evenly distributed across identities. The Indian group shows the widest range of variations per identity, followed by the Caucasian group. The African group covers a somewhat narrower range, while the Asian group displays the smallest range overall. The Indian and Caucasian groups exhibit a strong right skew, indicating that most identities are represented with relatively few images, while only a smaller number of the identities have a larger number of variations. The African group, although also skewed, shows a slightly more even spread across identities. By contrast, the Asian group stands out for being comparatively more balanced, with lower and higher counts of variations represented more evenly.



(a) Indian.

(b) Asian.

(c) Caucasian.

(d) African.

Figure 3.2: Distribution of the number of variations by ethnicity. The x-axis corresponds to the number of variations, while the y-axis represents the number of identities that have that number of variations.

Subset of BUPT-BalancedFace for comparison As we have mentioned previously, we faced computational limitations (see Section 3.2) when generating our synthetic datasets (baseline and ethnicity-specific IDiff-Face models), which made them relatively small (approximately 100 identities per ethnicity). Therefore, to evaluate our synthetic data in terms of photorealistic quality, identity separability and utility for training FR models, we constructed a reduced set from the BUPT-BalancedFace dataset, so that we would be able to have a fair comparison of the synthetic data with real data. Specifically, we sampled 500 distinct identities, taking 100 from each of the four ethnicity groups, preserving all of the images available for each selected identity.

3.4. Preprocessing: Face Alignment

Before embedding extraction, each RGB face image is aligned to the ArcFace geometry. We run an MTCNN detector (`keep_all`=True, `min_face_size`=1, `post_process`=False) on the current device to obtain candidate bounding boxes and their five-point landmarks

(eye centers, nose tip, mouth corners). If multiple faces are detected, we select the detection whose bounding-box center is closest to the image center, in order to choose the intended face in the presence of false detections.

Given the selected five landmarks $\mathbf{p} \in \mathbb{R}^{5\times 2}$, we estimate a 2D similarity transform that maps $\mathbf{p}$ to the ArcFace geometry $\mathbf{p}^\star$ (inter-ocular distance and pose), and warp the image accordingly to produce a 112×112 crop:

$$\mathcal{T}^\star \;=\; \arg\min_{\mathcal{T}\in \mathrm{Sim}(2)} \left\| \mathcal{T}(\mathbf{p}) - \mathbf{p}^\star \right\|_2^2, \qquad \tilde{x} \;=\; \mathrm{warp}(x; \mathcal{T}^\star).$$

If MTCNN fails to return landmarks (due to, for example, extreme pose, occlusion or low contrast), we do a deterministic resize to 112×112 without geometric normalization. To scale to large sets, file lists are sharded across devices with Lightning Fabric. Existing aligned files are skipped on subsequent runs, making the procedure idempotent. The result is a 112×112 ArcFace-aligned crop of the image that standardizes pose and scale, stabilizing downstream cosine-based comparisons with ElasticFace.

3.5. Extraction of the face embeddings

All embeddings were extracted with the ElasticFace loss [20], using an IResNet-100 network with a 512-dimensional feature head. We load the published checkpoint and switch the model to evaluation mode. Each RGB image is assumed to be a pre-aligned face crop. In order to extract face embeddings, the image is first loaded from disk and preprocessed to the input geometry expected by the backbone. Concretely, images are resized to 112×112 pixels, converted to tensors, and normalized channel-wise with `mean=0.5` and `std=0.5`, which maps pixel values to $[1, 1]$. The backbone f_θ maps an input x to a 512-D embedding $e = f_\theta(x)$, which we ℓ_2-normalize to enable cosine-based comparison:

$$\hat{e} \;=\; \frac{e}{\|e\|_2}.$$

3.6. Ethnicity classifier for the identity contexts

We trained a classifier using the pre-extracted BUPT-BalancedFace embeddings to predict the ethnicity of the uniform contexts we generated for sampling (process is described in Section 3.7). We used a simple multilayer perceptron (MLP) classifier due to the tabular nature and fixed-length of the embeddings. The model consists of a linear layer that maps the embedding dimension to 256, followed by a ReLU activation function and dropout regularization with rate 0.3 to mitigate overfitting. A final linear layer projects

the representation to the number of outputs, i.e., ethnicity classes, outputting logits for classification.

Training was done using an Adam optimizer with a learning rate of 0.0005. The loss function we are trying to minimize is the cross entropy. We used a batch size of 64 samples per iteration and a maximum of 100 epochs for training. To measure the model's performance, we performed a 80%-20% training-validation split. Early stopping was implemented with a patience of 10 epochs to stop the training if the validation loss did not improve. Performance metrics such as accuracy were monitored throughout the training, with a detailed evaluation using a confusion matrix and classification report.

3.7. Generation and classification of synthetic identity contexts

We generate random synthetic uniform identity context vectors, which will be used to condition our models on identity, as illustrated in Fig. 3.3. Each context is then classified into one of the four ethnicity groups using the ethnicity classifier in Section 3.6. We stop the generation when we have the same amount of identity contexts across ethnicity groups.

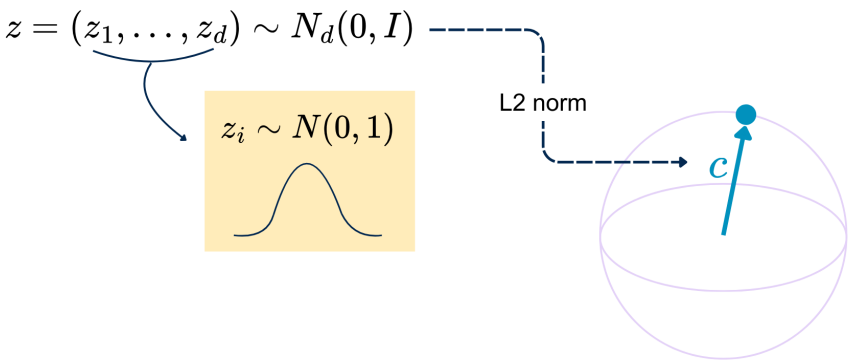


Figure 3.3: Generation of random uniform context vectors. These contexts are subsequently classified by ethnicity and used to guide the sampling process (see Section 2.5.1).

3.8. Expanding IDiff-Face: Our Ethnicity-Specific Pipeline

To evaluate the impact of implicit ethnicity conditioning on the quality and usefulness of synthetic faces, we implemented two complementary pipelines. The *baseline pipeline* follows the standard usage of IDiff-Face: a single model trained on the entire BUPT-BalancedFace dataset without any ethnicity conditioning, providing a reference point for all of the evaluation metrics. *Our proposed pipeline* trains four separate IDiff-Face models, one for each ethnicity group. The following subsections describe each pipeline in detail.

3.8.1 Baseline Approach: training a single IDiff-Face model

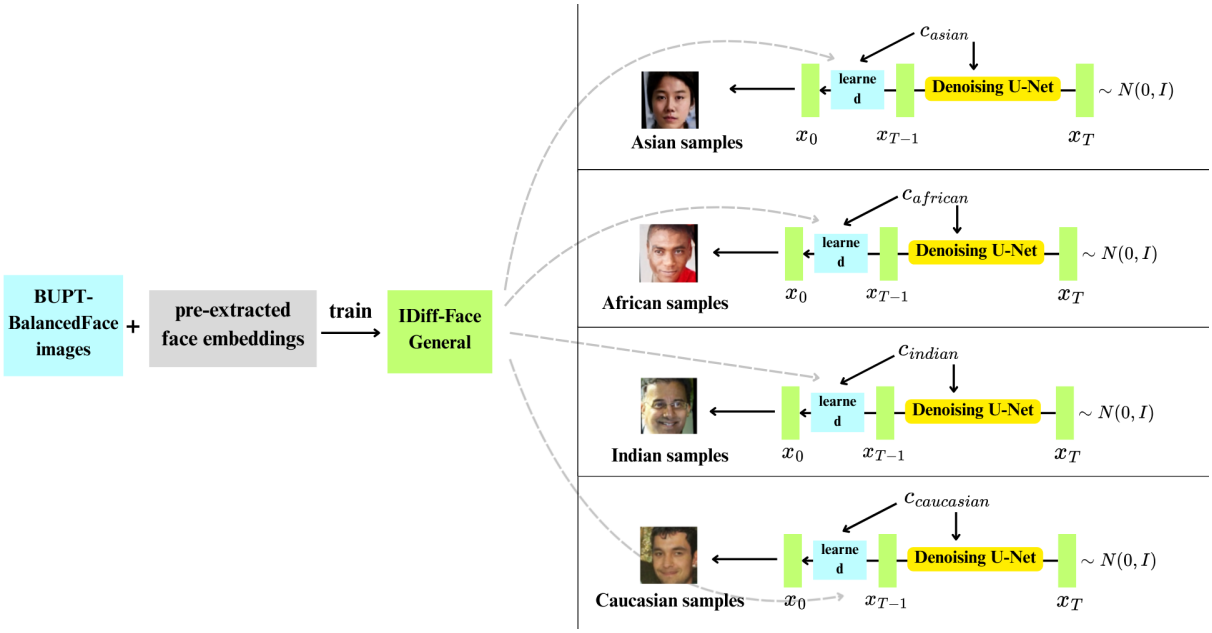
We train a single IDiff-Face model [12] on the entire BUPT-BalancedFace dataset [81] without any ethnicity conditioning. The dataset generated using this general IDiff-Face model is the synthetic baseline against which we can compare the one generated by our proposed ethnicity-specific approach. The pipeline, as we can see in Fig. 3.4a, consists of the following steps:

1. **Training of a general IDiff-Face model:** We train an IDiff-Face model with 25% CPD on the whole BUPT-BalancedFace dataset, as described in Section 3.9.
2. **Ethnicity-consistent sampling:** We take the identity-context vectors that we previously generated (read Section 3.7 for more information on the generation) and classified by ethnicity (check Section 3.6 to know more about the classifier we used). We use these ethnicity-labeled contexts to generate a synthetic dataset in which the generated samples are organized by ground-truth ethnicity.

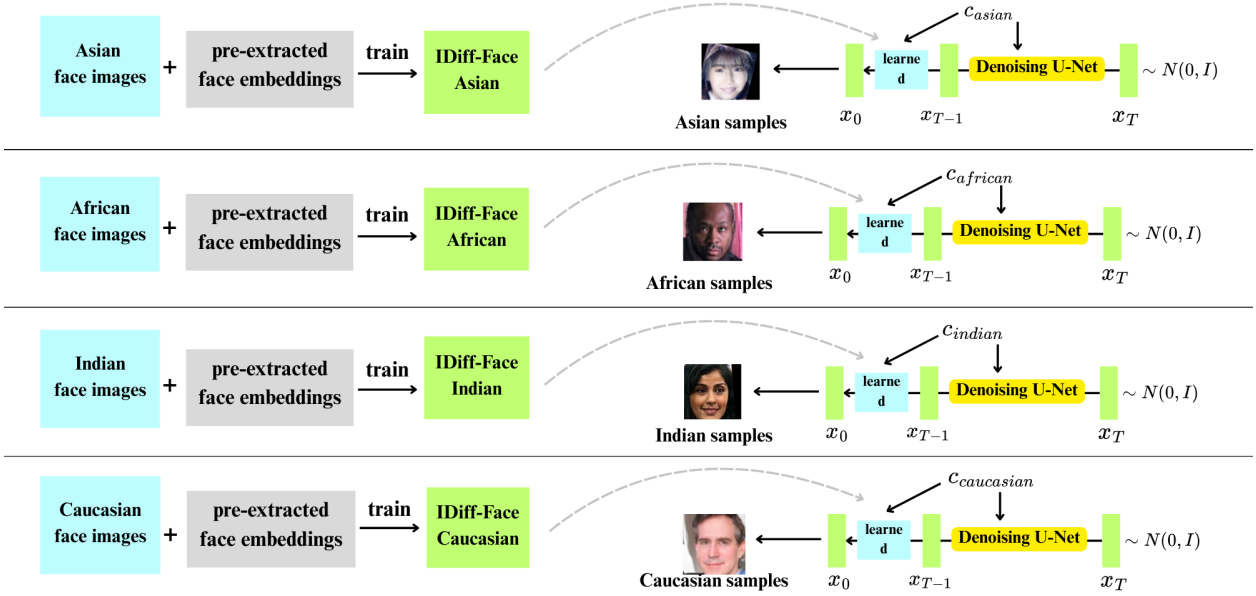
3.8.2 Our Approach: training ethnicity-specific IDiff-Face models

Next, we extend the IDiff-Face approach by training separate diffusion models for each ethnicity and conditioning the sampling process to ensure ethnicity consistency. The pipeline consists of the following steps (Fig. 3.4b):

1. **Training of ethnicity-specific models:** We train four IDiff-Face models with 25% CPD independently, each using only the images from one ethnicity group in the BUPT-BalancedFace dataset. The training is as described in Section 3.9.
2. **Ethnicity-consistent sampling:** We take the identity-context vectors that we previously generated and classified by ethnicity. Each set of ethnicity-labeled contexts is



(a) Schematic of the baseline IDiff-Face approach. We train an IDiff-Face model on the whole BUPT-BalancedFace dataset. This model is then used to generate face samples, where the denoising U-Net is conditioned on the corresponding demographic embedding (c_{asian}, $c_{african}$, c_{indian}, $c_{caucasian}$). Each row illustrates the generation process for one ethnicity, from Gaussian noise to a synthesized image.



(b) Schematic of the proposed ethnicity-specific IDiff-Face approach. We train separate ethnicity-specific IDiff-Face models. These models are used to generate face samples, where the denoising U-Net is conditioned on the corresponding demographic embedding (c_{asian}, $c_{african}$, c_{indian}, $c_{caucasian}$). Each row illustrates the generation process for one ethnicity, from Gaussian noise to a synthesized image.

Figure 3.4: Comparison between the baseline IDiff-Face approach (top) and the proposed ethnicity-specific IDiff-Face approach (bottom).

then used to sample from the corresponding IDiff-Face model, producing a synthetic dataset where the generated images are grouped by their ground-truth ethnicity.

3.9. Training the baseline and ethnicity-specific IDiff-Face models

In all experiments we trained a latent-diffusion variant of IDiff-Face with a DDPM noise schedule and a cross-attention U-Net backbone configured with 25% CPD and no context permutation. Unless stated otherwise, we used the same training seed, mixed-precision training, and batch size across all runs. All other training and optimization hyperparameters are listed in Appendix B.1. Every 50K epochs, we qualitatively evaluate the outputs of the models during training under various sampling scenarios by generating a grid of images: this allows us to do a structured comparison between unconditional, synthetic conditional and authentic conditional generations, taking as reference real images. We adopted this approach for validation during training in both the IDiff-Face approach and our ethnicity-specific approach. However, for our approach, instead of one, we do four grids corresponding to each ethnicity specific IDiff-Face model (African, Asian, Caucasian, Indian).

3.10. Evaluation of the synthetic face data

Firstly, in Section 3.10.1, we will evaluate the performance of the ethnicity classifier we trained, since it is a key component of our methodology (check the training details in Section 3.6). Next, in order to conduct a thorough evaluation of the synthetic facial data we generated using our ethnicity-specific approach, the samples will be evaluated in four aspects: *photorealism* (Section 3.10.2), *identity-separability* (Section 3.10.3), *utility in training FR* (Section 3.10.4) and *ethnicity consistency of the samples* (Section 3.10.5).

3.10.1 Evaluation of the ethnicity classifier

To evaluate the performance of the ethnicity classifier used for classifying embeddings into ethnicity groups, we compute the standard classification metrics of precision, recall and F1-score for each ethnic group c, as well as the overall classification accuracy (Equations 3.1). Here, $\text{TP}(c)$, $\text{FP}(c)$ and $\text{FN}(c)$ denote the number of true positives, false positives and false negatives for class c, respectively, and N denotes the total number of embeddings classified.

$$\begin{aligned}
\text{Precision}(c) &= \frac{\text{TP}(c)}{\text{TP}(c) + \text{FP}(c)} \\
\text{Recall}(c) &= \frac{\text{TP}(c)}{\text{TP}(c) + \text{FN}(c)} \\
\text{F1}(c) &= 2 \cdot \frac{\text{Precision}(c) \cdot \text{Recall}(c)}{\text{Precision}(c) + \text{Recall}(c)} \\
\text{Accuracy} &= \frac{\sum_c \text{TP}(c)}{N}.
\end{aligned} \tag{3.1}$$

3.10.2 Evaluation of the photorealistic image quality

With AI-generated images becoming increasingly common, with billions of images produced annually using text-to-image models, we need to have a metric that can objectively evaluate the photorealism of these images in a way that matches human perception. There are two types of metrics for evaluating photorealism: *pixel-level* comparison metrics and *model-based* metrics. We assess the photorealistic quality of the synthetic face images using well established metrics: *Structural Similarity Index Measure (SSIM)*, *Multiscale Structural Similarity Index Measure (MS-SSIM)*, *Fréchet Inception Distance (FID)*, *Inception Score (IS)*, and *Learned Perceptual Image Patch Similarity (LPIPS)*.

3.10.2.1 Metrics

Pixel-level comparison methods have long served as standard tools for assessing structural fidelity and quantifying noise variations between AI-generated images and their ground truth references. We will now talk about some of the most popular metrics of this sort.

Structural Similarity Index Measure (SSIM) The SSIM [82] is a perceptual metric used to evaluate the similarity between two images. Unlike pixel-wise comparisons such as Mean Squared Error (MSE) or Peak Signal-to-Noise Ratio (PSNR), which focus solely on absolute differences, SSIM considers structural information, luminance, and contrast. Let μ_x and μ_y be the mean intensities of images x (ground truth) and y (generated), representing luminance; σ_x^2 and σ_y^2 the variances in intensities of x and y, capturing contrast; σ_{xy} the covariance of x and y, capturing structural similarity; and c_1 and c_2 constants to prevent division by zero. Equation 3.2 gives us the value for SSIM. Considering the individual formulas for luminance $l(x,y)$, contrast $c(x,y)$ and structure $s(x,y)$, given by, respectively Equations 3.3, 3.4 and 3.5, Equation 3.2 is equivalent to Equation 3.6, where

the weights α, β, and γ are all set to 1. Researchers have shown SSIM to significantly outperform MSE and its derivates, such as PSNR, in accuracy [83]. However, SSIM still has several limitations when used for evaluating photorealism in AI generated images, for example, its sensitivity to small misalignments. Even if a generated image is slightly shifted, rotated, or misaligned compared to the reference, SSIM will penalize it heavily, even though it may still look realistic to the human perception. This becomes a problem in generative models like GANs or diffusion models, where the goal is not to match the reference pixel by pixel but rather to produce images that look natural.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{3.2}$$

$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1} \tag{3.3}$$

$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2} \tag{3.4}$$

$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}, \text{ where } c_3 = \frac{c_2}{2} \tag{3.5}$$

$$SSIM(x,y) = l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma \tag{3.6}$$

Multi-scale Structural Similarity Index Measure (MS-SSIM) The MS-SSIM [84] is a generalization of the SSIM, where, instead of computing the index at a single resolution (scale), it evaluates similarity across multiple scales by progressively downsampling the image and computing SSIM at each level. Let M be the number of scales, $l_M(x,y)$ the luminance similarity at the final scale, $c_j(x,y)$ and $s_j(x,y)$ the contrast and structure similarities at each scale, and α_j, β_j and γ_j the weights of each component. Then, the general form of MS-SSIM is given by Equation 3.7. The MS-SSIM is often better than SSIM in evaluating perceptual image quality. Since SSIM works at a single scale, it only considers local pixel-level similarities, whereas the multi-scale approach better mimics the human visual system, which perceives details at different resolutions.

$$MS-SSIM(x,y) = [l_M(x,y)]^{\alpha_M} \cdot \prod_{j=1}^{M} [c_j(x,y)]^{\beta_j} [s_j(x,y)]^{\gamma_j} \tag{3.7}$$

In contrast, model-based metrics rely on features extracted by pre-trained neural networks to assess how realistic, diverse and high-quality the images are.

Inception Score (IS) The IS was introduced in 2016 [85] and is calculated based on the output of a separate, pretrained Inception v3 image classification model applied to a sample of images generated by the generative model. For a given generated image x, the Inception v3 model predicts a probability distribution over the ImageNet classes $p_{dis}(\cdot|x)$, i.e., for a class label y, $p_{dis}(y|x)$ is the predicted probability distribution over class y (conditional distribution). The marginal distribution $\int p_{dis}(\cdot|x)p_{gen}(x)dx$ is the average of the conditional distributions over all generated images, representing the overall class distribution of the generated dataset. IS is calculated as the exponential of the KL divergence between the conditional and marginal distributions as described in Equation 3.8. A high IS indicates that the generated images are both high-quality and diverse: if the generated images are high quality, the Inception model will assign high probabilities to specific classes for each image (i.e., $p_{dis}(\cdot|x)$ will have low entropy); if the generated images are diverse, the marginal distribution $p_{dis}(y)$ will be spread across many classes (i.e., $p_{dis}(y)$ will have high entropy). As for its limitations, since IS only evaluates the generated images without comparing them to real images, it can be misleading if the generated images are unrealistic but diverse.

$$IS(p_{gen}, p_{dis}) = \exp(\mathbf{E}_{x\sim p_{gen}}[D_{KL}(p_{dis}(\cdot|x)|| \int p_{dis}(\cdot|x)p_{gen}(x)dx)]) \tag{3.8}$$

Fréchet Inception Distance (FID) Introduced in 2017 [86], the FID compares the distribution of the generated images with the distribution of a set of real images (”ground truth”). So, instead of evaluating individual images directly, it compares statistical representations derived from both the generated and reference image sets. Since diffusion models generate images with features from the ”ground truth” set, while still being different from any image in the training set, we can’t assess the quality of these models by direct pixel-wise comparison. Instead, the FID passes the two sets of images through a pretrained Inception v3 network, as if they were drawn from two multidimensional Gaussian distributions $\mathcal{N}(\mu, \Sigma)$ and $\mathcal{N}(\mu', \Sigma')$, and the distance between these two distributions is calculated as the earth mover’s distance (Fréchet distance) between the two Gaussian distributions, i.e., instead of pixel-wise comparison, the FID compares the mean and standard deviation of the deepest layer in Inception v3.Contrary to the IS, high FID scores indicate a poorer

generative model, with a score of 0 corresponding to a perfect model. In regards to the limitations of this metric, Chong and Forsyth [87] showed that it was statistically biased, meaning its expected value does not accurately reflect its true value. Moreover, despite FID being more consistent with human judgment than the IS, there are cases of inconsistency of this metric with human judgment [88]. Additionally, the FID metric, though inspired by the IS metric, is not meant replace it. Instead, the two metrics should be used together to comprehensively evaluate generated images. This is because classifiers that achieve the best (lowest) FID scores typically exhibit greater sample variety, while those achieving the best (highest) IS scores tend to produce higher-quality individual images. Using both metrics provides a more balanced assessment of both diversity and quality in generated outputs [89].

$$d_F(\mathcal{N}(\mu,\sigma),\mathcal{N}(\mu',\sigma'))^2 = ||\mu-\mu'||_2^2 + \text{tr}(\Sigma+\Sigma'-2(\Sigma\Sigma')^{\frac{1}{2}}) \tag{3.9}$$

Learned Perceptual Image Patch Similarity (LPIPS) The LPIPS, introduced in 2018 by Zhang et. al. [90], computes similarity through deep neural networks, like FID. This metric, unlike the ones mentioned previously, instead of assessing the quality of the generated image, measures perceptual similarity. Using pretrained CNNs (such as AlexNet or VGG), LPIPS extracts high-level perceptual features from images, measuring the distance in the network's feature space as a "perceptual loss" for image regression problems.

Metric	Type	Model	What it measures
SSIM	Pixel-based	–	How similar two images are in brightness, contrast and structure.
MS-SSIM	Pixel-based	–	Same as SSIM but computed at several image scales.
FID	Model-based	Inception v3	How close the generated images' features are to real images' features.
IS	Model-based	Inception v3	How confident and diverse the class predictions are.
LPIPS	Model-based	AlexNet, VGG	How similar the deep features of two images are: perceptual similarity of content.

Table 3.3: Simple overview of the metrics used to evaluate generated images.

3.10.2.2 Protocol

SSIM, MS-SSIM, FID, and LPIPS require a reference set of real images, for which we use the RFW dataset [80]. As a baseline, we replace the synthetic set with a subset of the

BUPT-BalancedFace dataset [81]. All metrics are computed with `TorchMetrics` [91] as follows:

- SSIM and MS-SSIM: SSIM is computed per image and MS-SSIM per batch, pairing each synthetic example (or batch) with its corresponding real data counterpar. We use a batch size of 32 and report the mean over the entire evaluation.
- LPIPS: we compute the learned perceptual distance between each synthetic batch and the matched real batch and report the aggregate mean.
- FID: we follow the standard two-pass protocol using Inception-v3 features (2048-D): real features are accumulated once, synthetic features are accumulated separately, and the Fréchet distance between the two Gaussian summaries is reported.
- IS: we run the Inception-v3 classifier on the synthetic images only and report the resulting scalar.

3.10.3 Evaluation of Identity Separability

We have mentioned previously that synthetic datasets should have both large intra-class diversity and large identity separability. Therefore, it is of extreme importance to assess these aspects of our data using quantitative metrics. We evaluate identity separability of the synthetic datasets we generated with the two approaches (the baseline IDiff-Face and our ethnicity-specific approach) by building genuine and impostor score distributions and analyzing them with cosine similarity–based metrics such as False Match Rate (FMR), False Non-Match Rate (FNMR), Equal Error Rate (EER) and Fisher Discriminant Ratio (FDR).

3.10.3.1 Metrics

FMR and FNMR are widely used for the evaluation of biometric verification systems, helping quantify the reliability of a system by measuring its susceptibility to:

- **FMR:** Incorrectly accepting an imposter (non-matching sample) as genuine.
- **FNMR:** Incorrectly rejecting a genuine sample.

Formally, for a similarity threshold τ

$$\text{FMR}(\tau) = \frac{\text{Number of false acceptances}}{\text{Total impostor comparisons}} \tag{3.10}$$

$$\text{FNMR}(\tau) = \frac{\text{Number of false rejections}}{\text{Total genuine comparisons}} \tag{3.11}$$

Equal Error Rate (EER) Represents the threshold for which the FMR and the FNMR are equal. Lower EER values indicate systems which overall discriminate better between genuine users and impostors.

Fisher's Discriminant Ratio (FDR) Quantifies the separability between classes in the feature space, measuring how discriminative a feature is for distinguishing two populations (e.g., genuine users vs. impostors). Rooted in Fisher's Linear Discriminant Analysis, FDR evaluates the ratio of between-class variance to within-class variance. Higher FDR values indicate stronger class separation, implying the feature is more effective for classification or verification tasks.

$$\text{FDR} = \frac{(\mu_G - \mu_I)^2}{\sigma_G^2 + \sigma_I^2} \tag{3.12}$$

3.10.3.2 Protocol

For the 25% CPD IDiff-Face configuration (see training details in Section 3.9), we load precomputed, ℓ_2-normalized embeddings $\{(e_i, y_i)\}$ from the aligned synthetic data (see image alignment details in Section 3.4), where y_i is the corresponding identity label and e_i is the 512-dimensional embedding vector for image i given by ElasticFace-R100 [20]. Since $\|e_i\|_2 = 1$, pairwise similarity is simply cosine similarity, computed as the dot product $s_{ij} = e_i^\top e_j \in [-1, 1]$.

We construct two score distributions: the genuine distribution includes all within-identity pairs obtained by taking the Cartesian product over indices with the same label (excluding self-pairs), and the impostor distribution consists of randomly sampled cross-identity pairs (with replacement) chosen so that the impostor set has the same size as the genuine set. With these balanced lists, we sweep a threshold t over $[-1, 1]$ to compute the false non-match rate $\text{FNMR}(t)$ from the genuine scores and the false match rate $\text{FMR}(t)$ from the

impostor scores, and report the EER at their intersection:

$$\mathrm{EER} \;=\; \mathrm{FNMR}(t^\star) \;=\; \mathrm{FMR}(t^\star), \qquad t^\star \;=\; \arg\min_t \Big|\, \mathrm{FNMR}(t) - \mathrm{FMR}(t) \,\Big|.$$

All statistics are computed using the Python library `pyeer` [92]. For transparency, the script saves histograms of the two cosine-score distributions and writes the raw genuine and impostor scores to disk.

Given the cosine score distributions for genuine pairs $G = \{s_k^G\}$ and impostor pairs $I = \{s_\ell^I\}$, we define $\mathrm{FMR}(t) = \Pr(s^I \geq t)$ and $\mathrm{FNMR}(t) = \Pr(s^G < t)$ for a decision threshold $t \in [-1, 1]$. For a target false-match threshold α, we find the smallest threshold t_α such that $\mathrm{FMR}(t_\alpha) \leq \alpha$ and report $\mathrm{FNMR}(t_\alpha)$ at that point. Following this convention, we denote:

$$\textbf{FMR10} \;=\; \mathrm{FNMR}(t_{0.10}) \qquad \text{(FNMR at } \mathrm{FMR} \leq 10\%\text{)},$$

$$\textbf{FMR20} \;=\; \mathrm{FNMR}(t_{0.05}) \qquad \text{(FNMR at } \mathrm{FMR} \leq 5\%\text{)},$$

$$\textbf{FMR100} \;=\; \mathrm{FNMR}(t_{0.01}) \qquad \text{(FNMR at } \mathrm{FMR} \leq 1\%\text{)}.$$

3.10.4 Evaluation of the utility of the images for training FR

We trained a ResNet-18 FR model with the ArcFace loss on our synthetic datasets (full hyper-parameters and augmentation settings are listed in Appendix B.2). Evaluation followed the 10-fold RFW protocol with mean $\pm$ std accuracy, TAR@FAR and XNorm reported.

3.10.5 Evaluation of Ethnicity Consistency

Synthetic faces belonging to the same identity should share consistent demographic attributes. We measure this by applying a high-accuracy ethnicity classifier trained on the BUPT-BalancedFace's embeddings to the ElasticFace embeddings of the synthetic dataset and computing the following statistics for each identity:

Majority-vote Accuracy (MVA) Let I be a synthetic identity with n_I generated images. Let c_i be the ethnicity predicted by the classifier for image i. The majority-vote accuracy for identity I corresponds to the proportion of its images that are classified into the ethnicity with the largest number of classifications for that identitity (Eq. 3.13). The overall MVA is given by Eq. 3.14, and corresponds to the average consistency per identity in the dataset.

If MVA $= 100\%$, then for each identity I in the dataset, $\text{MVA}_I = 100\%$ (every identity in the dataset are completely consistent).

$$\text{MVA}_I = \max_c \frac{1}{n_I} \sum_{i=1}^{n_I} \mathbb{1}[c_i = c] \tag{3.13}$$

$$\text{MVA} = \frac{1}{N} \sum_{I=1}^{N} \text{MVA}_I \tag{3.14}$$

Strict Consistency Rate (SCR) The fraction of identities for which all images are predicted as the same ethnicity is given by Eq. 3.15. If SCR = 1.0, then every identity in the dataset has all of its images with the same classification (every identity is completely consistent). If SCR is $x\%$, then $x\%$ of the identities in the dataset have all images classified into the same ethnicity group.

$$\text{SCR} = \frac{1}{N} \sum_{I=1}^{N} \mathbb{1} \left[\max_c \sum_{i=1}^{n_I} \mathbb{1}[c_i = c] = n_I \right]. \tag{3.15}$$

Intra-class Precision, Recall and F1-score For each ethnicity c, precision, recall and F1-score are computed by comparing the classifier's predictions with the target ethnicity used to condition the generation (Eq. 3.1).

4. Results and Discussion

In this chapter, we present and analyze the experimental results we obtained throughout this work. In Section 4.1, we begin by evaluating the performance of the classifier used to predict the ethnicity of the random synthetic uniform identity contexts, a key component in our methodology. In Section 4.2, we present and compare the samples generated by the two generative strategies: the baseline IDiff-Face strategy and our ethnicity-specific IDiff-Face approach. Finally, in Section 4.4, we evaluate and compare the generated datasets under four criteria aligned with our research questions: *photorealistic quality* (Section 4.4.1), *identity separability* (Section 4.4.2), *utility for training a small scale FR system* (Section 4.4.3) and *ethnicity consistency* (Section 4.4.4).

All experiments were conducted under limited computational resources, which restricted the volume of generated samples. These constraints are taken into account when interpreting the results presented in this chapter.

4.1. Ethnicity classifier of embeddings

As we can see in Table 4.1, the MLP trained on the ElasticFace embeddings from the BUPTBalancedFace dataset (check Section 3.6 for more details) achieved consistently high performance across all of the ethnicity groups (F1-scores ranging from 0.98 to 0.99). The Asian ethnicity group had the highest precision and recall (both 0.99). The remaining groups also achieved strong results, with the Indian having the slightly lowest recall (0.97), suggesting that a small portion of Indian samples were misclassified into other ethnic groups. The overall accuracy was of 98%, which shows a strong generalization ability of our classifier. In Appendix C, we present the confusion matrix, the training and evaluation losses and the training and evaluation accuracies of the ethnicity classifier.

Table 4.1: Performance of the ethnicity classifier evaluated on a validation set from the BalancedFace dataset. The table reports precision, recall, F1-score, and support for each class, as well as overall accuracy.

Class	Precision	Recall	F1-Score	Support
African	0.98	0.98	0.98	64,807
Asian	0.99	0.99	0.99	64,563
Caucasian	0.98	0.99	0.98	65,243
Indian	0.98	0.97	0.98	55,173
Accuracy		0.98		249,786

4.2. Sample quality comparison: baseline IDiff-Face vs. our ethnicity-specific extension

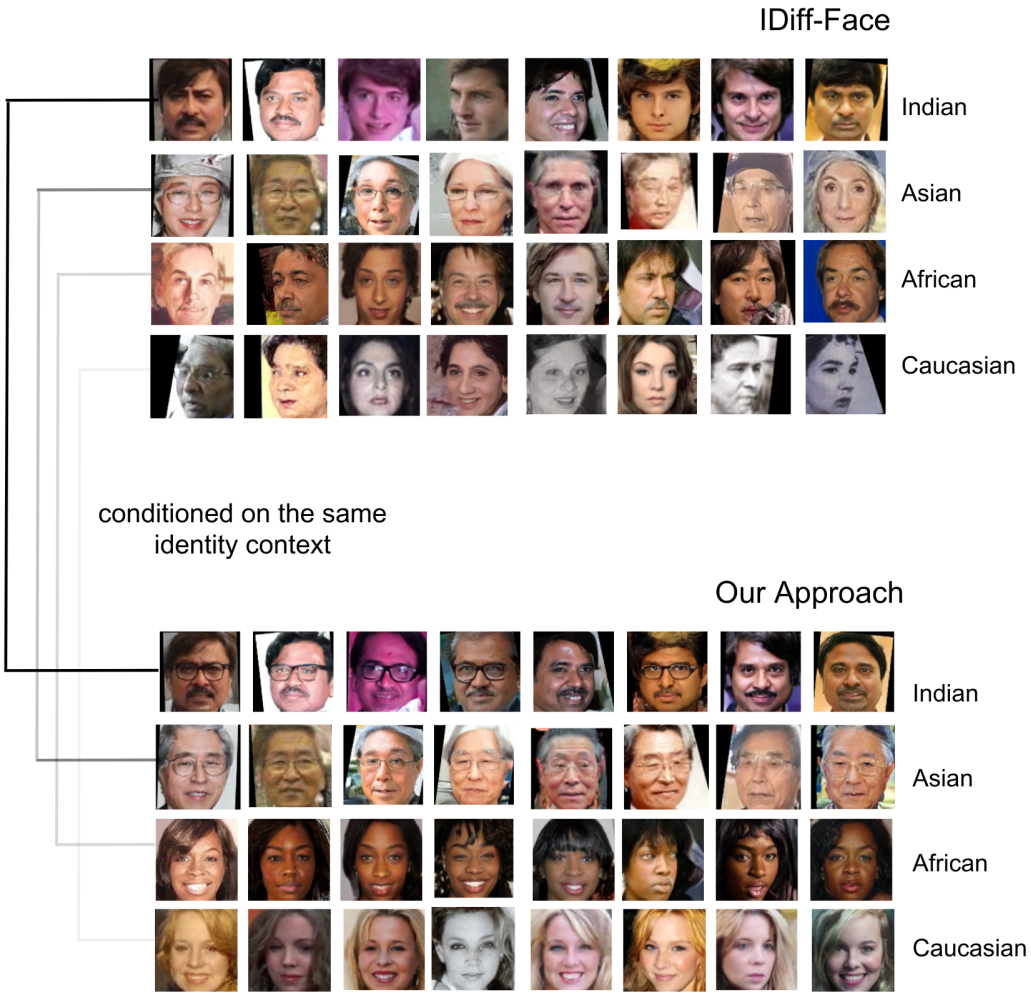


Figure 4.1: Examples of synthetic faces generated using the same identity context for four demographic groups. The top block shows samples from the IDiff-Face approach, while the bottom block shows samples from our approach. Each row consists of multiple generated variations of a single random uniform synthetic identity.

Figure 4.1 shows illustrative examples of identities generated by our ethnicity-specific IDiff-Face approach and by the baseline model. In this particular figure we deliberately selected one of the clearer cases to make the comparison easier to interpret.

In the samples generated using our ethnicity-specific IDiff-Face models (bottom grid), the

eyes, nose, jawline and overall proportions remain more or less consistent and ethnicity-specific characteristics, such as skin tone, are adapted in a way that appears both plausible and coherent with the respective ethnicity group (ethnicity consistency). We can also see natural variations in head pose, expression, lighting, hair styling and minor accessories that do not change or collapse the identity. Moreover, observing the figure, this approach appears to produce more stable identity cues across ethnicity groups than the baseline IDiff-Face model (top grid). The *improvement in terms of ethnicity consistency* is particularly clear in the Indian-identity examples (top row). Let us look at the third to seventh images generated by the baseline IDiff-Face model: we can see a noticeable shift toward Caucasian facial traits. This does not happen for the images generated for this identity when using our ethnicity-specific approach: they retain Indian facial characteristics much more consistently. Additionally, for the same African identity, the baseline IDiff-Face visually generates faces with non-African features, while our approach preserves African traits and the intended identity across all generated images.

4.3. Composition of the datasets (synthetic data and real dataset used for evaluation)

Table 4.2 serves as a summary for the composition of the three datasets used throughout our evaluation, broken down by ethnicity and total number of images. Both the synthetic datasets generated by our ethnicity-specific IDiff-Face models (“Ours”) and the baseline single-model IDiff-Face (“IDiff-Face”) were constructed to be approximately balanced across the four demographic groups (approximately 100 identities per ethnicity), and therefore results are not driven by obvious class-size differences.

Table 4.2: Number of images and identities of the datasets used for evaluation, by ethnicity and total. The real dataset corresponds to a subset of the BUPT-BalancedFace dataset. ”Ours” is the dataset generated using our approach, which consisted of training ethnicity-specific IDiff-Face models. ”IDiff-Face” corresponds to the baseline approach, where we train one single IDiff-Face model on the BUPT-BalancedFace dataset.

Dataset	**African**		**Asian**		**Caucasian**		**Indian**	
	Imgs	**#IDs**	**Imgs**	**#IDs**	**Imgs**	**#IDs**	**Imgs**	**#IDs**
Ours	4946	99	5000	100	5587	113	4960	100
IDiff-Face	4945	99	5000	100	5000	100	5000	100
Real data	4749	100	4583	100	4650	100	4348	100
Total Ours	20493 images							
Total IDiff-Face	19945 images							
Total Real data	18330 images							

4.4. Evaluation

We now proceed to the evaluation of the generated data using our ethnicity-specific approach. The goal of this section is to compare: (i) the synthetic dataset generated with our ethnicity-specific approach; (ii) the synthetic dataset generated with the baseline single-model IDiff-Face approach; (iii) real data.

4.4.1 Photorealistic quality

Table 4.3 shows the results of our photorealistic quality evaluation, comparing the ethnicity-specific IDiff-Face ("Ours") to the single-model baseline ("IDiff-Face") and the real data baseline.

Table 4.3: Photorealistic quality of samples generated with our approach and the baseline IDiff-Face compared to a subset of BUPT-BalancedFace (Real data), overall and by ethnicity. Best values between the two approaches are in bold with the relative change vs. the corresponding real data in parentheses. Arrows in the header indicate the preferred direction of the metric: ↓ lower is better, ↑ higher is better.

	Model-based			**Pixel-level**	
	LPIPS↓	FID↓	IS↑	SSIM↑	MS-SSIM↑
Overall					
Ours	0.423 (+1.7%)	96.4 (+499%)	1.065 (-1.6%)	0.317 (-8.1%)	**0.238** (-7.8%)
IDiff-Face	**0.422** (+1.4%)	**91.7** (+470%)	**1.066** (-1.5%)	**0.324** (-6.1%)	**0.238** (-7.8%)
Real data	0.416	16.1	1.082	0.345	0.258
African					
Ours	**0.418** (+2.0%)	**86.1** (+346%)	**1.066** (-1.4%)	**0.337** (-7.9%)	**0.243** (-8.6%)
IDiff-Face	0.422 (+2.9%)	99.3 (+415%)	**1.066** (-1.4%)	0.322 (-12.0%)	0.234 (-12.0%)
Real data	0.410	19.3	1.081	0.366	0.266
Asian					
Ours	**0.416** (-0.5%)	127.3 (+194%)	1.058 (-1.0%)	0.314 (-6.8%)	**0.250** (-6.7%)
IDiff-Face	0.421 (+0.7%)	**104.5** (+141%)	**1.065** (-0.4%)	**0.331** (-1.8%)	0.246 (-8.2%)
Real data	0.418	43.3	1.069	0.337	0.268
Caucasian					
Ours	0.421 (+5.2%)	**91.6** (+1209%)	**1.069** (-1.7%)	**0.354** (-9.2%)	**0.262** (-13%)
IDiff-Face	**0.420** (+5.0%)	100.0 (+1329%)	1.066 (-2.0%)	0.342 (-12.3%)	0.259 (-14.0%)
Real data	0.400	7.0	1.088	0.390	0.301
Indian					
Ours	**0.421** (+0.7%)	113.8 (+215%)	1.058 (-1.4%)	0.309 (-6.9%)	**0.262** (-7.7%)
IDiff-Face	0.422 (+1.0%)	**94.5** (+162%)	**1.067** (-0.6%)	**0.329** (-0.9%)	0.253 (-10.9%)
Real data	0.418	36.1	1.073	0.332	0.284

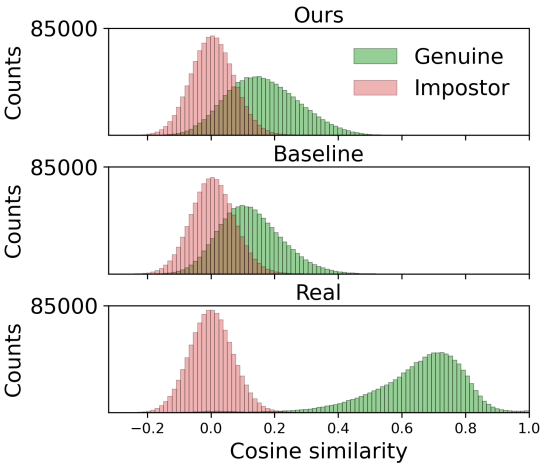
The synthetic face dataset generated with *our ethnicity-specific IDiff-Face approach overall underperforms the single-model baseline* in terms of photorealistic quality, as evidenced by the higher LPIPS and FID, lower IS and lower SSIM and MS-SSIM. However,

a closer analysis by ethnicity reveals that *our approach outperformed the single-model IDiff-Face for African and Caucasian samples.*

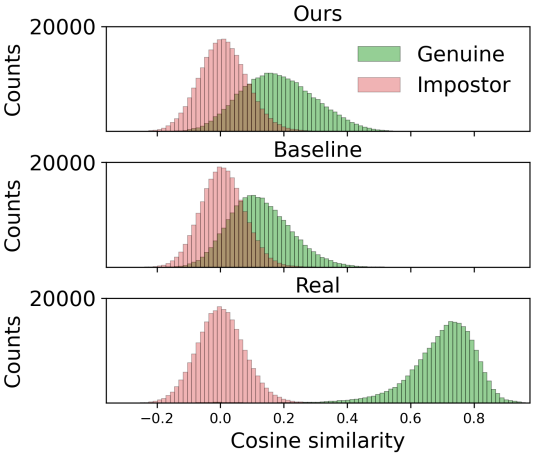
It is important to note that, overall, both synthetic datasets present: *an IS and LPIPS relatively close to that of the real data baseline*, indicating that the diversity and perceptual quality of the generated images is comparable to that of the real faces; *an SSIM and MS-SSIM within 8%*, which is desirable in this context as it indicates that the model is not producing images too similar to those of the training data; *a much larger FID* than the real dataset, with an increase of almost 500%, reflecting the distributional shift between synthetic and authentic samples. In the per-ethnicity breakdown, we see that, for both synthetic datasets, the *largest decrease in photorealistic quality* when compared to the real data baseline happens for the *Caucasian samples*.

4.4.2 Identity separability

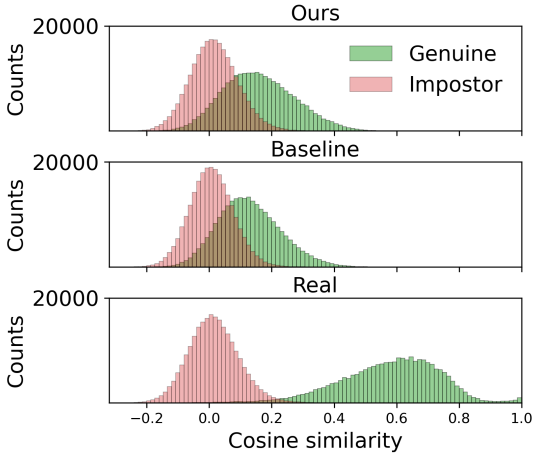
To assess whether the synthetic datasets present acceptable identity separability, we computed cosine similarity scores between all within-identity (genuine) and cross-identity (impostor) pairs of 512-D ElasticFace-R100 embeddings and plotted their distributions. For a well-separated dataset, the genuine distribution should be shifted to high similarity values and the impostor distribution to low similarity values, with minimal overlap. Figure 4.2 shows the genuine and impostor score distributions for our ethnicity-specific approach, the baseline IDiff-Face model, and a subset of real data, both overall and disaggregated by ethnicity. Table 4.4 summaries the corresponding error rates and distribution statistics.



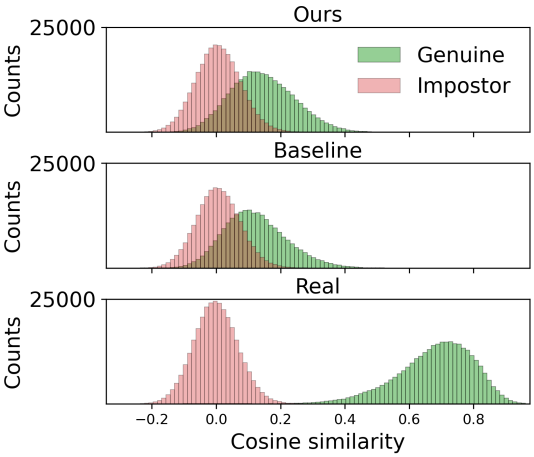
(a) Overall



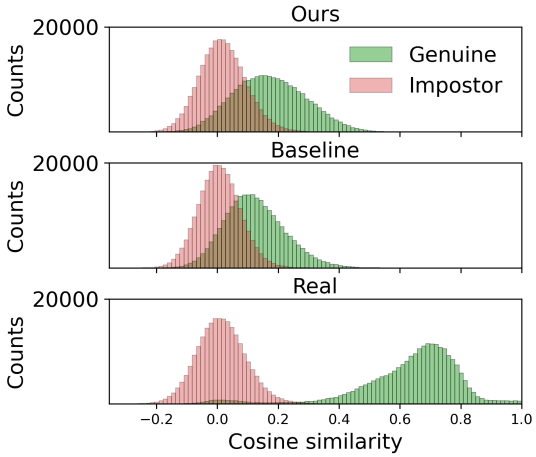
(b) African



(c) Asian



(d) Caucasian



(e) Indian

Figure 4.2: Genuine and impostor score distributions for genuine pairs and impostor pairs by ethnicity and overall for the different datasets (our ethnicity-specific IDiff-Face, the baseline IDiff-Face, and the subset from the BUPT-BalancedFace). The x-axis represents similarity scores between embeddings, while the y-axis indicates their frequency.

Table 4.4: Comparison of identity separability metrics across different ethnicity groups and overall using PyEER 0.5.6. Our approach, the baseline, and real data are shown. Arrows in the header indicate the preferred direction for each metric (↓: lower is better, ↑: higher is better). The x-axis of the histograms represents the similarity scores between two embeddings, while the y-axis represents their frequency of occurrence. A good separation is achieved when the genuine distribution (same identity) is shifted toward higher similarity scores, while the impostor distribution (different identities) remains concentrated at lower scores, with minimal overlap between the two.

	Operation metrics (%)				Score distributions				FDR ↑
Approach	**EER ↓**	**FMR100 ↓**	**FMR20 ↓**	**FMR10 ↓**	$\mu_G \uparrow$	$\sigma_G \downarrow$	$\mu_I \downarrow$	$\sigma_I \downarrow$	
Overall									
Ours	**19.5%**	**55.6%**	**37.2%**	**28.4%**	**0.163**	**0.112**	**0.006**	**0.069**	**1.44**
Baseline	24.3%	71.5%	51.7%	40.6%	0.124	0.098	0.007	0.069	0.97
Real data	1.29%	1.83%	1.10%	0.99%	0.650	0.149	0.002	0.069	5.58
African									
Ours	**18.6%**	**57.2%**	**36.9%**	**27.2%**	**0.182**	**0.115**	**0.010**	**0.076**	**1.57**
Baseline	24.3%	72.1%	52.3%	41.0%	0.124	0.097	0.007	0.069	0.96
Real data	0.49%	0.58%	0.42%	0.41%	0.699	0.110	0.002	0.076	7.36
Asian									
Ours	**22.9%**	**67.0%**	**47.2%**	**37.0%**	**0.159**	**0.113**	**0.016**	**0.076**	**1.10**
Baseline	23.9%	70.5%	50.5%	39.5%	0.129	0.099	0.008	0.069	1.00
Real data	1.72%	5.05%	2.12%	0.81%	0.578	0.158	0.015	0.080	4.50
Caucasian									
Ours	**21.3%**	**64.6%**	**44.2%**	**33.4%**	**0.143**	**0.100**	**0.007**	**0.070**	**1.26**
Baseline	24.8%	72.4%	52.7%	41.7%	0.122	0.098	0.007	0.068	0.93
Real data	0.48%	0.56%	0.46%	0.37%	0.677	0.126	0.0008	0.069	6.66
Indian									
Ours	**21.7%**	**63.2%**	**43.4%**	**33.6%**	**0.173**	**0.117**	**0.018**	**0.076**	**1.22**
Baseline	24.6%	72.3%	52.5%	41.3%	0.123	0.098	0.007	0.069	0.94
Real data	3.42%	4.73%	3.30%	3.06%	0.632	0.171	0.014	0.081	4.63

Across all of the ethnicity groups, the baseline IDiff-Face model shows substantial overlap between genuine and impostor curves, with a high overall EER (24.3%) and FMR100 (71.5%), indicating low discriminative power. The impostor distributions are well into the high-similarity region, so many impostor pairs produce scores similar to genuine pairs. When the model is trained separately for each ethnicity (our ethnicity-specific approach), the impostor curves shift slightly to the left (lower similarity) and the genuine curves stay at higher similarity, resulting in lower EER and FMR values in every ethnicity group. African identities benefit most from this change (EER 18.6%, FDR 1.57), showing a visibly cleaner separation of impostor and genuine scores. In contrast, Asian identities remain the most difficult to separate, with the highest EER (22.9%) and lowest FDR (1.10), reflecting a wider overlap between distributions. Although this ethnicity-specific training improves separability relative to the single-model baseline, the histograms also make clear that

our synthetic embeddings are still far from matching real data.

The same procedure applied to the real subset of BUPT-BalancedFace results in very low error rates (EERs below 3% across groups) and much higher FDR values, with impostor distributions tightly concentrated at low similarity and genuine pairs at high similarity. This visual contrast in Figure 4.2 shows how much more discriminative authentic embeddings remain compared with our synthetic ones. Nonetheless, it also reveals that separability varies across groups even for real data: Caucasian and African groups show the tightest (lowest variance) genuine distributions and the lowest impostor tails, whereas Asian and Indian groups have genuine distributions with slightly higher variance and somewhat higher impostor scores. Overall, these results confirm that our ethnicity-specific approach improves identity separability relative to the baseline IDiff-Face model but still lacks the discriminative strength of real facial images' embeddings.

4.4.3 Utility for training FR

All models were trained with ResNet-18 backbones on small, balanced datasets of only about 100 identities per ethnicity, which limits intra-class variation and naturally leads to lower verification accuracies on the challenging RFW benchmark (for more details on training, please check Section 3.10.4). The FR model trained on the synthetic data produced using the baseline single-model IDiff-Face converged after 18 epochs (best performance achieved at epoch 12); the FR model trained on the data generated with our ethnicity-specific IDiff-Face model was trained for 22 epochs (maximum validation accuracy at epoch 16); and the model trained on real data (subset of BUPT-BalancedFace) ran for 24 epochs (maximum accuracy at epoch 18).

Table 4.5 reports the mean XNorm, the standard deviation of the verification accuracy across the RFW folds (Std), and the highest verification accuracy obtained for each ethnicity. Despite the modest absolute values, the standard deviations are small (all under 3%), indicating stable behavior across folds.

Let us compare the performance of the FR system trained with the data generated using our approach with one generated using the baseline single-model IDiff-Face methodology. Our approach *outperformed IDiff-Face for Caucasian faces* (accuracy of 67.6%) *and African faces* (accuracy of 56.9%). However, it *underperformed IDiff-Face for Asian and Indian identities*. Across all training regimes, *African faces consistently presented*

Table 4.5: Verification performance on the RFW dataset per ethnicity. We compare ResNet-18 models trained on data generated by the baseline single-model IDiff-Face ("IDiff-Face"), our ethnicity-specific extension ("Ours"), and real data (BalancedFace subset). Metrics shown are XNorm, is the standard deviation of the verification accuracy across the RFW folds (Std), and verification accuracy at the epoch with highest accuracy. Bold indicates the best performance per metric.

Ethnic group	Approach	XNorm	Std	Accuracy
African	IDiff-Face	19.34	2.6%	56.2%
	Ours	19.70	1.7%	**56.9%**
	Real data	19.40	2.1%	57.6%
Asian	IDiff-Face	19.79	1.7%	**64.9%**
	Ours	19.58	1.6%	64.0%
	Real data	19.45	1.9%	64.4%
Indian	IDiff-Face	19.98	1.8%	**64.7%**
	Ours	19.94	2.2%	64.2%
	Real data	19.80	1.8%	64.9%
Caucasian	IDiff-Face	19.32	1.6%	67.5%
	Ours	19.11	2.2%	**67.6%**
	Real data	19.09	2.0%	67.8%
Overall	IDiff-Face	–	–	**63.2%**
	Ours	–	–	63.1%
	Real data	–	–	63.6%

lower verification accuracy, with a gap of more than 10% in accuracy relative to the best-performing group. Moreover the FR model trained on the synthetic datasets achieve *verification accuracies that are comparable to those obtained with the real dataset baseline*.

These results highlight that the proposed approach yields competitive verification performance relative to the baseline and to real data, while also showcasing the difficulty of achieving a balanced performance across ethnicities in FR systems.

4.4.4 Ethnicity consistency

We evaluated how well each synthetic dataset preserves the specified ethnicity using our high-accuracy classifier applied to the ElasticFace embeddings. Tables 4.6 and 4.7, respectively, show the evaluation results at the identity and image levels; Fig. 4.3 shows the corresponding confusion matrices for the ethnicity-classifier's performance on the synthetic data produced by both approaches.

As shown in Table 4.6, applying the ethnicity classifier to the embeddings produced by our ethnicity-specific IDiff-Face yields strict consistency for 11.7 % of identities, compared with

Table 4.6: Ethnicity consistency comparison between the baseline IDiff-Face approach (399 IDs) and our approach (412 IDs).

Approach	Maj. vote Acc.	Strict Cons.	Strict+Corr.	Strict+Wrong	Mean Mode Frac.
Baseline	0.368	0.003	0.000	0.003	0.661
Ours	**0.995**	**0.117**	**0.117**	**0.000**	**0.889**

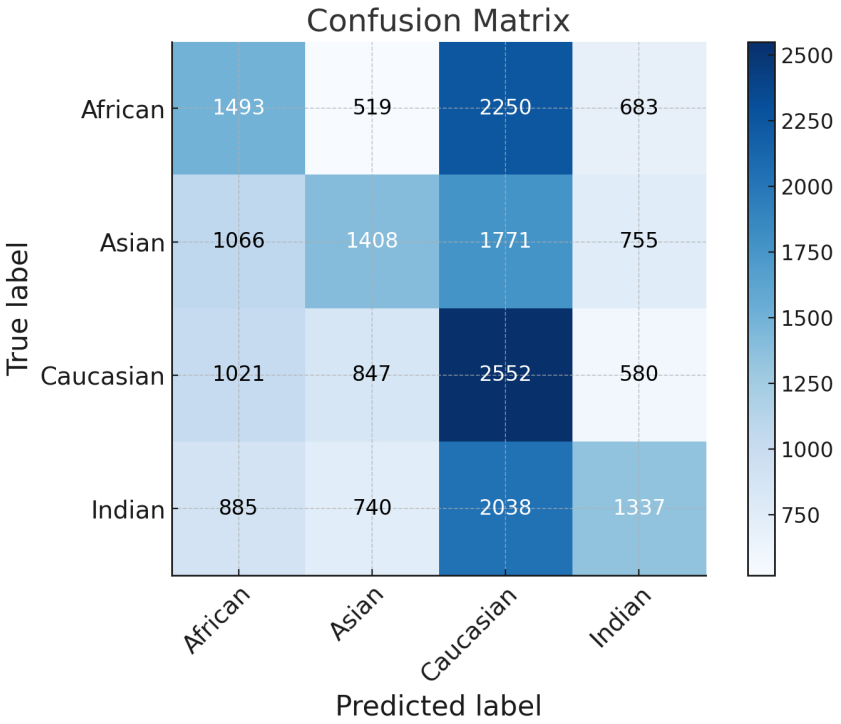
Table 4.7: Performance of the ethnicity classifier we trained when applied to the embeddings of the synthetic datasets generated with our approach and with the IDiff-Face baseline.

Approach	Class	Precision	Recall	F1	Support
Our approach	African	0.886	0.826	0.855	4 946
	Asian	0.948	0.928	0.938	5 000
	Caucasian	0.865	0.935	0.898	5 587
	Indian	0.854	0.853	0.854	4 960
	Accuracy		0.887		20 493
	Macro avg	0.888	0.885	0.886	–
	Weighted avg	0.888	0.887	0.887	–
IDiff-Face	African	0.334	0.302	0.317	4 945
	Asian	0.401	0.282	0.331	5 000
	Caucasian	0.296	0.510	0.375	5 000
	Indian	0.399	0.267	0.320	5 000
	Accuracy		0.340		19 945
	Macro avg	0.358	0.340	0.336	–
	Weighted avg	0.358	0.340	0.336	–

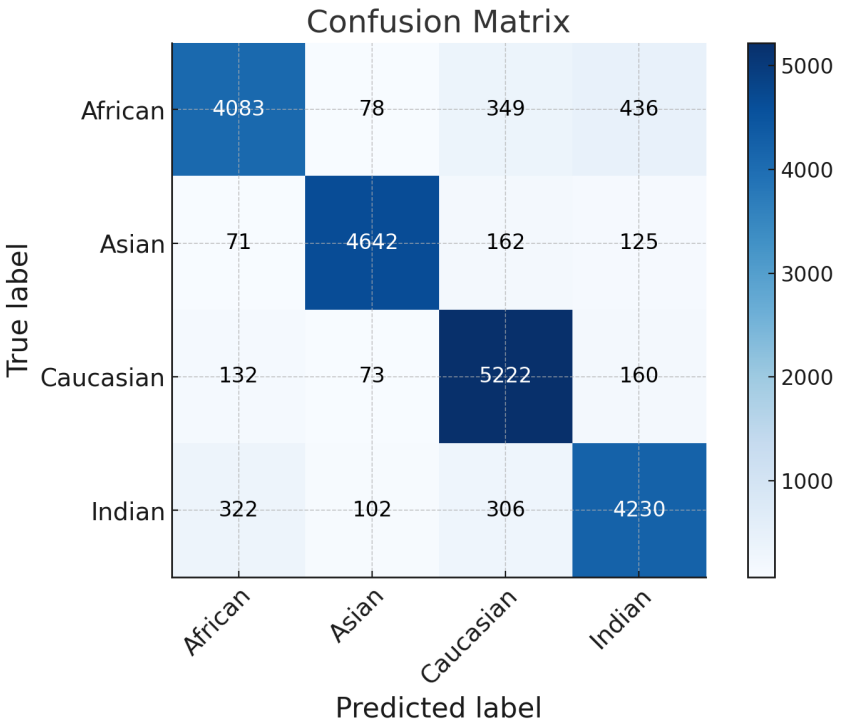
only 0.03 % for the synthetic baseline. The mean mode fraction also indicates markedly stronger internal consistency (88.9 % versus 66.1 %), and the overall classification accuracy rises to 88.7 % versus 34.0 % for the baseline. For both approaches the macro and weighted averaged scores are very close, reflecting the near-balanced class supports and indicating that results are not dominated by a single class.

The confusion matrices in Fig. 4.3 illustrate these differences visually. In the baseline model (Fig. 4.3a), African, Asian and Indian embeddings are frequently misclassified as Caucasian and Caucasian embeddings are often misclassified as African, whereas these cross-ethnicity confusions are much less pronounced in our approach.

These results show that *our ethnicity-specific IDiff-Face produces embeddings with ethnicity attributes that are preserved far more faithfully and consistently than those from the single-model IDiff-Face baseline*; however, even for the embeddings produced by the ethnicity-specific approach, there are still variations in terms of consistency across ethnicities: *Asian and Caucasian samples achieve higher consistency than African and Indian samples.*



(a) Baseline single-model IDiff-Face.



(b) Our ethnicity-specific approach.

Figure 4.3: Confusion matrices of the ethnicity classifier applied to the synthetic datasets. (a) Baseline single-model IDiff-Face approach. (b) Our ethnicity-specific approach. Rows correspond to ground-truth ethnicities (African, Asian, Caucasian, Indian) and columns to predicted ethnicities. The color intensity represents the number of embeddings per cell.

5. Conclusions and Future Work

Motivated by recent privacy regulations, this thesis explored the potential of diffusion-based generative models in generating synthetic face datasets that are both high-quality and privacy-preserving. With this goal in mind, we extended an existing latent diffusion-based framework by training four ethnicity-specific models on the corresponding subsets of a balanced real dataset. Using these models, we generated a synthetic dataset comprising roughly 100 identities per ethnicity, each with about 50 variations. We then compared this dataset against (i) a synthetic dataset of similar size sampled from a single baseline model trained on the entire real dataset, and (ii) a subset of the real dataset containing a comparable number of identities per ethnicity. The evaluation focused on four research questions:

RQ1. How does conditioning a diffusion model on ethnicity affect the photorealistic quality of synthetic faces compared to a single general model?

Training ethnicity-specific models did not improve overall photorealistic quality relative to the single-model baseline. Across the full dataset, the baseline is slightly better on all computed metrics. However, our ethnicity-specific approach outperformed the single-model baseline for African and Caucasian samples. Despite these improvements, there remains a large distributional shift between synthetic and authentic samples.

RQ2. Does training ethnicity-specific diffusion models improve identity separability relative to a single unconditioned model?

Training ethnicity-specific models consistently improved identity separability for all ethnicity groups, with the improvement being most pronounced for African samples. Nonetheless, the gap to real data remains very large.

RQ3. Can synthetic face datasets produced by diffusion models support small-scale face recognition training with performance approaching that of real data and does training ethnicity-specific diffusion models reduce demographic imbalance in face recognition performance?

When training a small-scale FR system on the synthetic dataset generated with our ethnicity-specific approach, verification results on validation data were very similar to those obtained from FR trained on the baseline synthetic dataset and only slightly below the accuracies from real data. African faces remain the hardest group to verify, regardless of training data. These findings indicate that under this small-scale regime, our specialized synthetic data is competitive with the baseline approach and with the real data, but does not yet solve the performance imbalance across ethnicity groups.

RQ4. To what extent does ethnicity conditioning improve the consistency of ethnicity cues in generated embeddings?

Compared to the baseline single-model approach, our ethnicity-specific approach generated synthetic faces whose embeddings were far more consistent with their intended ethnicity labels. Most identities now have their images classified in the same ethnicity group, and classification accuracies increase substantially relative to the single-model baseline. Asian and Caucasian embeddings are easiest to recognize, while African and Indian remain hardest, though still outperforming by far the single-model baseline. Taken together, these results show that ethnicity-specific generation improved the consistency of ethnicity cues in the synthetic dataset, although variation in performance across groups persists and the imbalance in FR accuracy is not yet resolved.

In addition to the quantitative evaluations, the qualitative sample quality comparison reinforces our findings. Visually, the ethnicity-specific models produced synthetic faces that maintained more stable identity cues and displayed more coherent ethnicity-specific characteristics than the baseline single-model approach. This visual evidence complements the metric-based analysis and strengthens the conclusion that conditioning on ethnicity can enhance ethnicity fidelity without degrading overall image quality.

Moreover, this work makes two main contributions: (i) it establishes a reproducible pipeline for training and evaluating ethnicity-conditioned diffusion models for face data generation; (ii) it advances the discussion on privacy-preserving biometrics by demonstrating how generative models can be adapted to address demographic imbalance and regulation constraints.

In conclusion, this thesis demonstrates the potential of diffusion models for generating synthetic face data conditioned on ethnicity. Though the results demonstrate promise, there are still several directions open for future research.

Future work. First, an important direction is the scaling up of synthetic datasets. Due to computational limitations, this work focused on producing a representative subset of samples. Larger-scale experiments would allow for more robust training of face recognition systems exclusively on synthetic data and provide a clearer assessment of whether synthetic datasets can fully replace real ones. Second, improvements in photorealistic quality are necessary, since there remains a considerable gap between synthetic and real data in terms of fidelity. Future work could explore hybrid approaches that combine diffusion models with adversarial refinement or use more advanced latent architectures to bridge this gap. Finally, additional conditioning factors beyond ethnicity could be integrated. Attributes such as age, gender, pose, or lighting conditions would provide richer variability and enable the generation of datasets that more closely mirror the complexity of real-world data. Exploring multi-attribute conditioning could also help address fairness challenges.

References

[1] L. I. Millett and J. N. Pato, *Biometric recognition: Challenges and opportunities*. National Academies Press, 2010. [Cited on page 1.]

[2] K. Wang, J. I. Tamir, A. D. Goyeneche, U. Wollner, R. Brada, S. Yu, M. Lustig, and N. K. Wang, "High fidelity deep learning-based mri reconstruction with instance-wise discriminative feature matching loss high fidelity deep learning-based mri reconstruction with instance-wise discriminative feature matching loss submited to magnetic resonance in medicine," *Magnetic Resonance in Medicine*, vol. 88, no. 1, pp. 476–491, apr 2022. [Online]. Available: https://doi.org/10.1002/mrm.29227 [Cited on page 1.]

[3] D. Yi, Z. Lei, S. Liao, and S. Z. Li, "Learning face representation from scratch," 11 2014. [Online]. Available: https://arxiv.org/abs/1411.7923v1 [Cited on page 1.]

[4] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, "Ms-celeb-1m: A dataset and benchmark for large-scale face recognition," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9907 LNCS, pp. 87–102, 7 2016. [Online]. Available: https://arxiv.org/abs/1607.08221v1 [Cited on page 1.]

[5] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman, "Vggface2: A dataset for recognising faces across pose and age," *Proceedings - 13th IEEE International Conference on Automatic Face and Gesture Recognition, FG 2018*, pp. 67–74, 10 2017. [Online]. Available: https://arxiv.org/abs/1710.08092v2 [Cited on page 1.]

[6] European Parliament and the Council of the European Union, "Regulation (eu) 2016/679 (general data protection regulation) — article 9: Processing of special categories of personal data," Official Journal of the European Union, L 119, p. 1–88, May 2016, sets conditions for processing special categories of personal data, including biometric data used for uniquely identifying a person. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj [Cited on page 1.]

[7] European Parliament and the Council of the European Union, "Regulation (eu) 2024/1689 of the european parliament and of the council of 13 june 2024 laying down harmonised rules on artificial intelligence (artificial intelligence act) — article

5(1)(e),” Official Journal of the European Union, L series, Jul. 2024, prohibits AI systems that create or expand facial recognition databases through the untargeted scraping of facial images from the internet or CCTV footage. [Online]. Available: http://data.europa.eu/eli/reg/2024/1689/oj [Cited on page 1.]

[8] K. Peng, A. Mathur, and A. Narayanan, “Mitigating dataset harms requires stewardship: Lessons from 1000 papers,” *arXiv preprint arXiv:2108.02922*, 2021. [Cited on page 1.]

[9] P. Grother, M. Ngan, and K. Hanaoka, “Face recognition vendor test (frvt) part 3: Demographic effects,” National Institute of Standards and Technology, Tech. Rep. NISTIR 8280, Dec. 2019. [Online]. Available: https://pages.nist.gov/frvt/reports/demographics/nistir_8280.pdf [Cited on page 1.]

[10] P. Grother, “Frvt part 8: Summarizing demographic differentials,” National Institute of Standards and Technology, Tech. Rep. NISTIR 8429 (interim), Jul. 2022. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ir/2022/NIST.IR.8429.ipd.pdf [Cited on page 1.]

[11] K. Kärkkäinen and J. Joo, “Fairface: Face attribute dataset for balanced race, gender, and age for bias measurement and mitigation,” in *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2021, pp. 1547–1557. [Cited on page 1.]

[12] F. Boutros, J. H. Grebe, A. Kuijper, and N. Damer, “Idiff-face: Synthetic-based face recognition through fizzy identity-conditioned diffusion models,” *Proceedings of the IEEE International Conference on Computer Vision*, pp. 19 593–19 604, 2023. [Cited on pages 1, 3, 29, 30, 32, 33, 35, and 42.]

[13] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2022-June, pp. 10 674–10 685, 12 2021. [Online]. Available: https://arxiv.org/abs/2112.10752v2 [Cited on pages 1, 3, 14, and 15.]

[14] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, 12 2013. [Online]. Available: https://arxiv.org/abs/1312.6114v11 [Cited on pages 3, 5, and 6.]

[15] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *Science Robotics*, vol. 3, pp. 2672–2680, 6 2014. [Online]. Available: https://arxiv.org/abs/1406.2661v1 [Cited on pages 3, 5, and 8.]

[16] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, “Deep unsupervised learning using nonequilibrium thermodynamics,” in *Proceedings of the 32nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 37. PMLR, 2015, pp. 2256–2265. [Online]. Available: http://proceedings.mlr.press/v37/sohl-dickstein15.html [Cited on pages 3, 6, 9, and 14.]

[17] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, “Arcface: Additive angular margin loss for deep face recognition,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699. [Cited on pages 3, 19, 20, 25, and 27.]

[18] H. Wang, Y. Wang, Z. Zhou, X. Ji, D. Gong, J. Zhou, Z. Li, and W. Liu, “Cosface: Large margin cosine loss for deep face recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5265–5274. [Cited on pages 3, 20, 23, and 25.]

[19] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, “Sphereface: Deep hypersphere embedding for face recognition,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [Cited on pages 3, 20, 22, 23, and 25.]

[20] F. Boutros, N. Damer, F. Kirchbuchner, and A. Kuijper, “Elasticface: Elastic margin loss for deep face recognition,” *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, vol. 2022-June, pp. 1577–1586, 9 2021. [Online]. Available: https://arxiv.org/abs/2109.09416v4 [Cited on pages 3, 19, 20, 25, 27, 32, 40, 50, and 92.]

[21] A. L. Samuel, “Some studies in machine learning using the game of checkers,” *IBM Journal of Research and Development*, vol. 3, no. 3, pp. 210–229, 1959. [Cited on page 4.]

[22] A. van den Oord, N. Kalchbrenner, L. Espeholt, K. Kavukcuoglu, O. Vinyals, and A. Graves, “Conditional image generation with pixelcnn decoders,” in *Advances in Neural Information Processing Systems 29*, 2016, pp. 4790–4798. [Cited on page 5.]

[23] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," in *Arxiv*, 2016. [Online]. Available: https://arxiv.org/abs/1609.03499 [Cited on page 5.]

[24] D. Rezende and S. Mohamed, "Variational inference with normalizing flows," in *International conference on machine learning*. PMLR, 2015, pp. 1530–1538. [Cited on page 5.]

[25] C. Viroli and G. J. McLachlan, "Deep gaussian mixture models," *Statistics and Computing*, vol. 29, no. 1, pp. 43–51, 2019. [Cited on page 5.]

[26] S. Mohamed and B. Lakshminarayanan, "Learning in implicit generative models," *arXiv preprint arXiv:1610.03483*, 2016. [Cited on page 5.]

[27] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Proceedings of The 33rd International Conference on Machine Learning*. PMLR, 2016, pp. 1558–1566. [Cited on page 6.]

[28] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=PxTIG12RRHS [Cited on pages 6 and 11.]

[29] J. M. Joyce, *Kullback-Leibler Divergence*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 720–722. [Online]. Available: https://doi.org/10.1007/978-3-642-04898-2_327 [Cited on page 7.]

[30] Y. Kossale, M. Airaj, and A. Darouichi, "Mode collapse in generative adversarial networks: An overview," in *2022 8th International Conference on Optimization and Applications (ICOA)*, 2022, pp. 1–6. [Cited on page 9.]

[31] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241. [Cited on pages 12, 13, and 32.]

[32] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020. [Cited on page 14.]

[33] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” *arXiv preprint arXiv:2010.02502*, 2020. [Cited on page 14.]

[34] P. Esser, R. Rombach, and B. Ommer, “Taming transformers for high-resolution image synthesis,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 12 868–12 878, 12 2020. [Online]. Available: https://arxiv.org/abs/2012.09841v3 [Cited on page 15.]

[35] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991. [Cited on page 16.]

[36] B. Moghaddam, W. Wahid, and A. Pentland, “Beyond eigenfaces: probabilistic matching for face recognition,” in *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, 1998, pp. 30–35. [Cited on page 16.]

[37] J. H. Lai, P. C. Yuen, and G. C. Feng, “Face recognition using holistic fourier invariant features,” *Pattern Recognition*, vol. 34, no. 1, pp. 95–109, 2001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320399002009 [Cited on page 16.]

[38] W. Deng, J. Hu, and J. Guo, “Extended src: Undersampled face recognition via intraclass variant dictionary,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, pp. 1864–1870, 2012. [Cited on page 16.]

[39] L. Zhang, M. Yang, and X. Feng, “fairfacentation or collaborative representation: Which helps face recognition?” in *2011 International Conference on Computer Vision*, 2011, pp. 471–478. [Cited on page 17.]

[40] C. Liu and H. Wechsler, “Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition,” *IEEE Transactions on Image Processing*, vol. 11, pp. 467–476, 4 2002. [Cited on page 17.]

[41] T. Ahonen, A. Hadid, and M. Pietikäinen, “Face description with local binary patterns: Application to face recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 2037–2041, 2006. [Cited on page 17.]

[42] Z. Cao, Q. Yin, X. Tang, and J. Sun, "Face recognition with learning-based descriptor," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2707–2714, 2010. [Online]. Available: https://www.researchgate.net/publication/224164307_Face_recognition_with_learning-based_descriptor [Cited on page 17.]

[43] T. H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma, "Pcanet: A simple deep learning baseline for image classification?" *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, vol. 24, pp. 5017–5032, 12 2015. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/26340772/ [Cited on page 17.]

[44] Z. Lei, M. Pietikainen, and S. Z. Li, "Learning discriminant face descriptor," *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, pp. 289–302, 2 2014. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/24356350/ [Cited on page 17.]

[45] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled Faces in the Wild: A Database forStudying Face Recognition in Unconstrained Environments," in *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*. Marseille, France: Erik Learned-Miller and Andras Ferencz and Frédéric Jurie, Oct. 2008. [Online]. Available: https://inria.hal.science/inria-00321923 [Cited on page 17.]

[46] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012. [Cited on page 17.]

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. [Cited on page 17.]

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9. [Cited on page 17.]

[49] 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*, 2016, pp. 770–778. [Cited on page 17.]

[50] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, “Deepface: Closing the gap to human-level performance in face verification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708. [Cited on page 17.]

[51] F. Schroff, D. Kalenichenko, and J. Philbin, “Facenet: A unified embedding for face recognition and clustering,” in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2015, pp. 815–823. [Cited on pages 17 and 27.]

[52] Y. Sun, Y. Chen, X. Wang, and X. Tang, “Deep learning face representation by joint identification-verification,” *Advances in neural information processing systems*, vol. 27, 2014. [Cited on page 17.]

[53] G. B. Huang and E. Learned-Miller, “Labeled faces in the wild: Updates and new reporting procedures,” *Dept. Comput. Sci., Univ. Massachusetts Amherst, Amherst, MA, USA, Tech. Rep*, vol. 14, no. 003, 2014. [Cited on page 17.]

[54] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, “Toward open set recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 7, pp. 1757–1772, 2013. [Cited on page 17.]

[55] F. Boutros, V. Struc, J. Fierrez, and N. Damer, “Synthetic data for face recognition: Current state and future prospects,” *Image and Vision Computing*, vol. 135, p. 104688, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0262885623000628 [Cited on pages 26 and 28.]

[56] H. Qiu, B. Yu, D. Gong, Z. Li, W. Liu, and D. Tao, “Synface: Face recognition with synthetic data,” 8 2021. [Online]. Available: https://arxiv.org/abs/2108.07960v2 [Cited on pages 27, 29, and 34.]

[57] F. Boutros, M. Huber, P. Siebke, T. Rieber, and N. Damer, “Sface: Privacy-friendly and accurate face recognition using synthetic data,” *2022 IEEE International Joint Conference on Biometrics, IJCB 2022*, 6 2022. [Online]. Available: https://arxiv.org/abs/2206.10520v1 [Cited on pages 27, 29, and 34.]

[58] G. Bae, M. D. L. Gorce, T. Baltrusaitis, C. Hewitt, D. Chen, J. Valentin, R. Cipolla, and J. Shen, “Digiface-1m: 1 million digital face images for face recognition,” *Proceedings - 2023 IEEE Winter Conference on Applications of Computer Vision, WACV 2023*, pp. 3515–3524, 10 2022. [Online]. Available: https://arxiv.org/abs/2210.02579v1 [Cited on pages 27, 29, and 34.]

[59] F. Boutros, M. Klemt, M. Fang, A. Kuijper, and N. Damer, "Unsupervised face recognition using unlabeled synthetic data," *2023 IEEE 17th International Conference on Automatic Face and Gesture Recognition, FG 2023*, 11 2022. [Online]. Available: https://arxiv.org/abs/2211.07371v1 [Cited on pages 27 and 29.]

[60] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker, "Towards large-pose face frontalization in the wild," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 4010–4019, 4 2017. [Online]. Available: https://arxiv.org/abs/1704.06244v3 [Cited on page 28.]

[61] Y. Shen, P. Luo, J. Yan, X. Wang, and X. Tang, "Faceid-gan: Learning a symmetry three-player gan for identity-preserving face synthesis," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 821–830, 12 2018. [Cited on page 28.]

[62] Y. Deng, J. Yang, D. Chen, F. Wen, and X. Tong, "Disentangled and controllable face image generation via 3d imitative-contrastive learning," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 5153–5162, 4 2020. [Online]. Available: https://arxiv.org/abs/2004.11660v2 [Cited on page 28.]

[63] M. Kim, F. Liu, A. Jain, and X. Liu, "Dcface: Synthetic face generation with dual condition diffusion model," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12 715–12 725, 4 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:258170376 [Cited on pages 29 and 30.]

[64] M. Yeung, T. Teramoto, S. Wu, T. Fujiwara, K. Suzuki, and T. Kojima, "Variface: Fair and diverse synthetic dataset generation for face recognition," *CoRR*, vol. abs/2412.06235, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2412.06235 [Cited on page 29.]

[65] B. Biesseck, P. Vidal, L. Coelho, R. Granada, and D. Menotti, "Tcdiff: Triple condition diffusion model with 3d constraints for stylizing synthetic faces," *arXiv preprint arXiv:2409.03600*, 2024. [Online]. Available: https://arxiv.org/abs/2409.03600 [Cited on page 29.]

[66] Y. Mi, Z. Zhong, Y. Huang, Q. Yuan, X. Zhao, J. Xu, S. Ding, S. Wang, R. Guo, and S. Zhou, "Data synthesis with diverse styles for face recognition via 3dmm-guided

diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025, pp. 21 203–21 214. [Cited on page 29.]

[67] E. Caldeira, N. Damer, and F. Boutros, "Negfacediff: The power of negative context in identity-conditioned diffusion for synthetic face generation," *arXiv preprint arXiv:2508.09661*, 2025. [Cited on pages 29 and 31.]

[68] M. A. H. Reaj, R. D. Gupta, M. Y. Rahat, N. Fahad, M. J. Hasan, and T. H. Liew, "Gandiff fr: Hybrid gan diffusion synthesis for causal bias attribution in face recognition," *arXiv preprint arXiv:2508.11334*, 2025. [Cited on pages 29 and 30.]

[69] T. Karras, M. Aittala, S. Laine, E. Härkönen, J. Hellsten, J. Lehtinen, and T. Aila, "Alias-free generative adversarial networks," in *Proc. NeurIPS*, 2021. [Cited on page 29.]

[70] H. Wu, J. Singh, S. Tian, L. Zheng, and K. W. Bowyer, "Vec2face: Scaling face dataset generation with loosely constrained vectors," in *2025 International Conference on Learning Representations*, 2025. [Online]. Available: https://arxiv.org/abs/2409.02979v3 [Cited on page 30.]

[71] H. Wu, J. Singh, S. Tian, L. Zheng, and K. Bowyer, "Vec2face+ for face dataset generation," *arXiv preprint arXiv:2507.17192*, 2025. [Cited on page 30.]

[72] P. Shahreza and S. Marcel, "Hyperface: Generating synthetic face recognition datasets by exploring the face embedding hypersphere," *arXiv preprint arXiv:2411.08470*, 2024. [Online]. Available: https://arxiv.org/abs/2411.08470 [Cited on page 30.]

[73] M. Cobbinah, H. Nunoo-Mensah, P. Ebenezer Adjei, F. Adoma Acheampong, I. Acquah, E. Tutu Tchao, A. Selasi Agbemenu, J. John Kponyo, and E. Abaidoo, "Diversity in stable gans: A systematic review of mode collapse mitigation strategies," *Engineering Reports*, vol. 7, no. 6, p. e70209, 2025. [Cited on page 30.]

[74] Y. Zou, Y. Wang, and X. Lu, "Auto-encoding generative adversarial networks towards mode collapse reduction and feature representation enhancement," *Entropy*, vol. 25, no. 12, p. 1657, 2023. [Cited on page 30.]

[75] H. Rosenberg, S. Ahmed, G. Ramesh, K. Fawaz, and R. K. Vinayak, "Limitations of face image generation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 13, 2024, pp. 14 838–14 846. [Cited on page 31.]

[76] P. Esser, R. Rombach, and B. Ommer, “Taming transformers for high-resolution image synthesis,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12 873–12 883. [Cited on pages 32, 88, and 89.]

[77] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, “Labeled faces in the wild: A database forstudying face recognition in unconstrained environments,” in *Workshop on faces in’Real-Life’Images: detection, alignment, and recognition*, 2008. [Cited on page 34.]

[78] 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. [Cited on page 34.]

[79] S. Sengupta, J. C. Cheng, C. D. Castillo, V. M. Patel, R. Chellappa, and D. W. Jacobs, “Frontal to profile face verification in the wild,” in *IEEE Conference on Applications of Computer Vision*, February 2016. [Cited on page 34.]

[80] M. Wang, W. Deng, J. Hu, X. Tao, and Y. Huang, “Racial faces in-the-wild: Reducing racial bias by information maximization adaptation network,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 692–702. [Cited on pages 36 and 48.]

[81] M. Wang and W. Deng, “Mitigating bias in face recognition using skewness-aware reinforcement learning,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9322–9331. [Cited on pages 36, 38, 42, and 49.]

[82] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, 4 2004. [Online]. Available: https://www.researchgate.net/publication/3327793_Image_Quality_Assessment_From_Error_Visibility_to_Structural_Similarity [Cited on page 45.]

[83] L. Zhang, L. Zhang, X. Mou, and D. Zhang, “A comprehensive evaluation of full reference image quality assessment algorithms,” *Proceedings - International*

Conference on Image Processing, ICIP, pp. 1477–1480, 12 2012. [Online]. Available: https://research.polyu.edu.hk/en/publications/a-comprehensive-evaluation-of-full-reference-image-quality-assess [Cited on page 46.]

[84] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multi-scale structural similarity for image quality assessment," *Conference Record of the Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 1398–1402, 2003. [Online]. Available: https://www.researchgate.net/publication/4071876_Multiscale_structural_similarity_for_image_quality_assessment [Cited on page 46.]

[85] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, X. Chen, and X. Chen, "Improved techniques for training gans," *Advances in Neural Information Processing Systems*, vol. 29, 2016. [Online]. Available: https://github.com/openai/improved-gan. [Cited on page 47.]

[86] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in Neural Information Processing Systems*, vol. 30, 2017. [Cited on page 47.]

[87] M. J. Chong and D. Forsyth, "Effectively unbiased fid and inception score and where to find them," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 6069–6078. [Cited on page 48.]

[88] S. Liu, Y. Wei, J. Lu, and J. Zhou, "An improved evaluation framework for generative adversarial networks," *ArXiv*, vol. abs/1803.07474, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4377863 [Cited on page 48.]

[89] J. Ho, "Classifier-free diffusion guidance," *ArXiv*, vol. abs/2207.12598, 2022. [Online]. Available: https://api.semanticscholar.org/CorpusID:249145348 [Cited on page 48.]

[90] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595. [Cited on page 48.]

[91] N. S. Detlefsen, J. Borovec, J. Schock, A. H. Jha, T. Koker, L. Di Liello, D. Stancl, C. Quan, M. Grechkin, and W. Falcon, "Torchmetrics - measuring reproducibility in

pytorch," *Journal of Open Source Software*, vol. 7, no. 70, p. 4101, 2022. [Online]. Available: https://doi.org/10.21105/joss.04101 [Cited on page 49.]

[92] M. A. Martínez, "Pyeer: Biometric systems performance evaluation in python," 2023, python package. Accessed 8 Sep 2025. [Online]. Available: https://pypi.org/project/pyeer/ [Cited on page 51.]

[93] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, 1943. [Cited on page 79.]

[94] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958. [Cited on page 79.]

[95] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org. [Cited on pages 79, 80, and 83.]

[96] J. Han and C. Morag, "The influence of the sigmoid function parameters on the speed of backpropagation learning," in *From Natural to Artificial Neural Computation*, ser. Lecture Notes in Computer Science, J. Mira and F. Sandoval, Eds., vol. 930. Springer, 1995, pp. 195–201. [Cited on page 80.]

[97] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf [Cited on page 86.]

[98] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens, "Stand-alone self-attention in vision models," in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Cited on page 87.]

[99] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, and J. Uszkoreit, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv*, June 2021. [Cited on pages 87 and 88.]

A. Background Knowledge and State-of-the-art

A.1. Neural Networks

The Artificial Neural Network (ANN) is the foundation of Machine Learning, allowing models to mimic the way biological neurons in the human brain identify patterns, weigh options and come to a decision.

The core idea behind neural networks emerged in the mid-20th century, with early models like the perceptron, which was introduced in concept for the first time in 1943 by Warren McCulloch and Walter Pitts [93] and formally implemented by Frank Rosenblatt in 1958 [94], with the aim of replicating decision-making processes.

Modern neural networks evolved to handle high-dimensional data and complex functions through layered architectures, and are widely used in several areas such as classification, regression, and pattern recognition.

An ANN consists of a set of interconnected processing units called *neurons*, typically organized into *layers*: the first layer (input); the intermediate layers, or hidden layers (it is not obligatory for these layers to exist, but they generally do); and the last layer (ouput). Neurons belonging to one layer connect only to neurons of the previous and next layers.

The biggest advantage in using neural networks to learn patterns in the data over traditional models such as linear regression, is that *neural networks are able to learn non-linear, complex patterns from the input data*. But how are neural networks able to capture these complex patterns?

An ANN adds an *activation function* for its neurons, which computes the output of the neuron based on its inputs and respective weights. A non-linear activation function introduces *non-linearity* into the model: without it, a neural network would simply behave similarly to a linear regression model [95]. We will now explore how the value for each neuron in each layer of the network is computed.

The neuron p from the network takes as input a vector $\mathbf{x}^{(p-1)} = (x_1^{(p-1)}, ..., x_n^{(p-1)})^T$, consisting of the n neurons from the previous layer connected to neuron p. Consider $\mathbf{w}_p = (w_{1p}, ..., w_{np})$ the vector containing the weights associated with each connection, b the bias vector. The linear combination of these parameters S_p, also known as its activation state, is given by Equation A.1.

$$S_p = f(\mathbf{x}^{(p-1)}) = \mathbf{w}^T\mathbf{x}^{(p-1)} + b \tag{A.1}$$

Let $f_{\text{activation}}$ be the activation function of the network. The value z taken by the neuron i.e., the prediction made by the model is given by Equation A.2.

$$\mathbf{z} = f_{\text{activation}}(f(\mathbf{x}^{(p-1)})) \tag{A.2}$$

A.1.1 Popular activation functions

Below, we explore some of the most popular activation functions: the sigmoid function, the tanh function, and the ReLU function [95].

- **The sigmoid function**

  This function, characterized by its ”S” shape, is continuous and differentiable, making it essential for gradient-based optimization methods. Its output ranges from 0 to 1, making it particularly useful for classification problems [96].

  $$\sigma(x) = \frac{1}{1+\exp(-x)} \tag{A.3}$$

  However, the sigmoid exhibits a very large gradient when the inputs are between -2 and 2, which can cause significant changes in the output for very close inputs. For centered data, typically, the hyperbolic tangent (tanh) function is preferred in order to avoid getting caught in slow learning regions during the training process.

- **The hyperbolic tangent (tanh) function**

  The tanh function is a shifted version of the sigmoid, with a zero centered output (outputs in $[-1, 1]$).

  $$\tanh x = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} = \frac{2}{1+\exp(-2x)} - 1 = 2 \cdot \sigma(2x) - 1 \tag{A.4}$$

- **The Rectified Linear Units (ReLU) function**

  The ReLU activation function outputs 0 for negative inputs, and the value of the input itself for positive ones:

  $$\text{ReLU}(x) = \max\{0, x\} \tag{A.5}$$

Both the sigmoid and the tanh functions can be affected by vanishing gradients, which makes ReLU preferable in Deep Neural Networks.

Training a neural network consists of adjusting its weights and biases to minimize the error between predictions and ground-truth.

The training starts with a forward pass, where the input vector is passed through the network. Each layer has as output $\mathbf{z}^{(p)} = f_{\text{Activation}}(\mathbf{W}^{(p)}\mathbf{x}^{(p-1)} + \mathbf{b}^{(p)})$. This process continues layer by layer until the final output (prediction) is produced.

Then, the network's prediction $\hat{\mathbf{y}}$ is compared with the ground-truth $\mathbf{y}$ using a loss function $L(\hat{\mathbf{y}}, \mathbf{y})$, such as the mean squared error (classification) or the cross-entropy (classification).

A.1.2 Optimization

Optimization is usually done using *backpropagation*, in which the network propagates back the error using the gradient of the loss function. Once these gradients are computed, an optimization algorithm, such as *Stochastic Gradient Descent (SGD)*, is used in order to adjust the weights and biases in the direction that minimizes the loss (objective).

Let η be a *learning rate*, which is an hyperparameter that determines the step size during the update. Let L be a loss function. When optimizing using SGD, the weights w are updated as described in Equation A.6.

$$w := w - \eta \frac{\partial L}{\partial w} \tag{A.6}$$

While it is the most common approach to optimize the neural network's weights, SGD has some inherent limitations due to being *largely dependent on the gradients of the loss function*. For convex functions, SGD will easily find the global minimum. However, it can struggle to do so with non-convex functions since the process stops as the gradient approaches zero, being possible to get stuck at local minimums or saddle points. A way to mitigate this issue is to *introduce noise* to the gradient.

Additionally, if the gradients get too small, enough so that the precision of the machine is not enough to differentiate them from zero, the learning process slows down and may even stop: *vanishing gradients*. Analogously, if the gradients get too large, it may cause

the training to become unstable, with the weights changing too drastically: *exploding gradients*.

A.1.3 Types of hidden layers

The structure of the hidden layers in a network is incredibly relevant to how well the network can capture the underlying structure in the data. We will now present some important layer structures.

- **Fully connected layers**

  If every neuron in one layer is connected to every neuron in the following layer, we say that the layers are *fully connected*, or *dense*.

- **Convolutional layers**

  Kernels, also known as *convolutional filters*, are small matrices (for example, 3×3 or 5×5) of weights used to *detect patterns* such as textures, edges and shapes in the input image.

  The convolutional filter slides over the input matrix, performing convolutions using the *Frobenius inner product* in order to generate a feature map, which will contribute to the input of the next layer.

  As the name suggest, *convolutional layers* are layers that apply convolution operations to the input. They constitute the primary blocks of CNN. The size of the kernel is a hyperparameter that affects the network's behavior.

- **Pooling layers**

  A *pooling layer* downsamples and aggregates the information from the previous layer, *removing redundant information*, *reducing the computational cost* of the network, and *making the model more robust* to small changes in the input.

  Consider an image represented by a tensor $x \in \mathbb{R}^{H \times W \times C}$, where H is the height, W is the width, and C is the number of channels ($C = 3$ for RGB images). A pooling layer outputs a tensor $y \in \mathbb{R}^{H' \times W' \times C'}$. Consider two variables: f, the "filter size", corresponding to the dimensions of the filter, and s, the "stride" of the filter, corresponding to how many units the filter moves across the image at each step. The receptive field of an entry in the output tensor are the entries in the input that are able to affect that entry.

Max Pooling (MaxPool) is commonly used in CNNs to perform dimensionality reduction of feature maps. It first calculates the maximum value of each region covered by the filter (patch) and assigns this value to the corresponding position in the output tensor, as described in Equation A.7.

$$y_k = \max(x_{k'} : k' \text{ in the receptive field of } k) \tag{A.7}$$

Similarly, in *Average Pooling (AvgPool)*, instead of taking the maximum value of each patch, we take the average of the values in the patch (Equation A.8).

$$y_k = \text{average}(x_{k'} : k' \text{ in the receptive field of } k) \tag{A.8}$$

A.1.4 Feed Forward Network (FFN)

The FFN is the simplest type of ANN [95], where the *information flow is one directional*, starting from the input layer, going through the hidden layers, and finishing at the ouput layer. FFNs with multiple hidden layers are commonly referred to as *Multi-layer perceptron (MLP)*.

Usually, the hidden layers are *fully connected*. They cannot include any cycles or loops since the information in the network only flows forward, and can therefore be seen as an acyclic fully connected weighed graph.

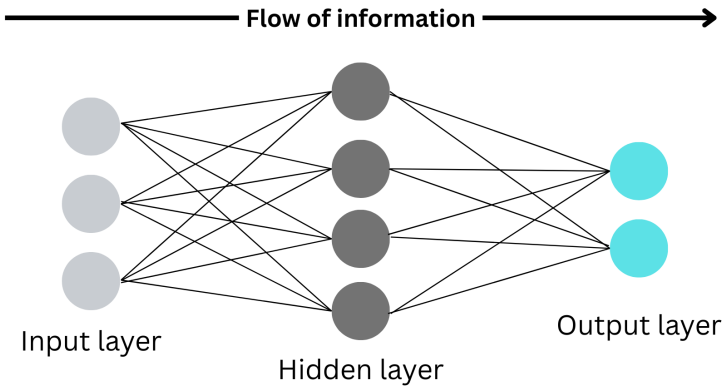


Figure A.1: Schematic representation of an FFN. Information flows unidirectionally from the input layer, through one or more hidden layers, to the output layer. Each neuron applies a weighted sum followed by a non-linear activation function, enabling the network to learn complex patterns from data.

A.1.5 Convolutional Neural Network (CNN)

Standard FFNs do not scale well to images: in an FFN, every input neuron connects to every hidden neuron, which implies that processing an image of size $224 \times 224 \times 3$ (~ 150K pixels) would require the optimization of millions of weights in the first layer alone, making the optimization of the network computationally inefficient and prone to overfitting. Additionally, FFNs do not make use of spacial information, ignoring the fact that nearby pixels are correlated and form meaningful local patterns (edges, textures, shapes).

CNNs address these limitations by introducing architectural constraints that act as a form of regularization. Instead of learning individual weights for each pixel location, CNNs use *weight sharing*, meaning that the same kernel is applied across the entire image. These networks also use local connectivity, where each neuron is only connected to a small neighborhood of the input, instead of the whole image. Pooling operations reduce the dimensionality of the feature maps, further constraining how the network represents information.

The CNN consists of an input layer, a set of hidden layers and an output layer. The hidden layers include one or more convolutional layers. The remaining layers are usually pooling layers, fully connected layers, and normalization layers.

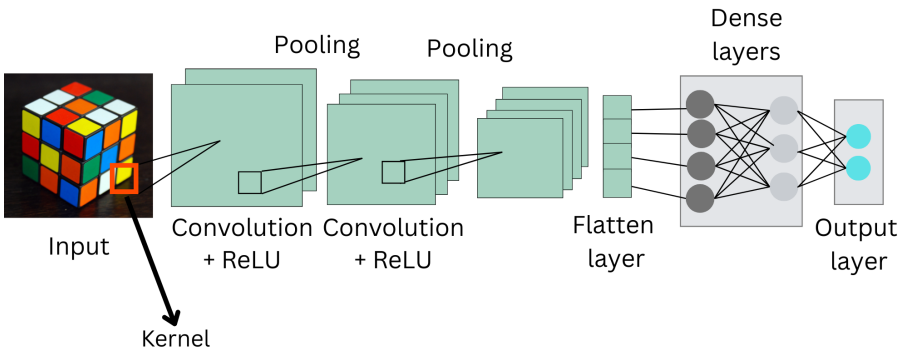


Figure A.2: Schematic representation of a CNN. A convolutional filter (kernel) is applied to the input image to extract local features. The resulting feature maps are passed through a ReLU activation, which sets negative values to zero and preserves positive ones, introducing non-linearity. A pooling operation (such as max pooling) is then applied to reduce dimensionality, lower computational cost, and increase robustness to small variations in the input. Repeated convolution+ReLU+pooling operations output progressively smaller feature maps that capture increasingly abstract features. Finally, the feature maps are flattened into a vector and processed by fully connected layers that combine the extracted features to output the final classification.

A.2. Stochastic processes, Markov Chains and Stationarity

A *stochastic process* is a collection of random variables $\{X_t\}_{t\in T}$ indexed by a set T (often time). Formally, for each $t \in T$, X_t is a random variable defined on a common probability space $(\Omega, \mathcal{F}, \mathbb{P})$ with state space $\mathcal{X}$. We can think of a stochastic process as a random function $t \mapsto X_t(\omega)$ describing the evolution of some quantity over time under uncertainty. If $T = \{0, 1, 2, \dots\}$ the process is called *discrete-time*, and if $T \subseteq \mathbb{R}$ it is called *continuous-time*.

A (discrete–time) Markov chain is a stochastic process $\{X_t\}_{t\geq 0}$ on a state space $\mathcal{X}$ such that

$$\mathbb{P}(X_{t+1} = x \mid X_t = x_t, \dots, X_0 = x_0) = \mathbb{P}(X_{t+1} = x \mid X_t = x_t), \tag{A.9}$$

i.e., the conditional distribution of the next state depends only on the current state. This conditional distribution is specified by a *transition kernel* $q(x|x')$, so that

$$\mathbb{P}(X_{t+1} \in A \mid X_t = x') = \int_A q(x|x')\, dx, \tag{A.10}$$

for any measurable set $A \subseteq \mathcal{X}$.

Given an initial distribution $p_0(x) = \mathbb{P}(X_0 = x)$, the distribution at time t is

$$p_t(x) = \int q(x|x')\, p_{t-1}(x')\, dx', \quad t \geq 1. \tag{A.11}$$

A probability distribution $\pi(x)$ on $\mathcal{X}$ is called *stationary* for the Markov chain if

$$\pi(x) = \int q(x|x')\, \pi(x')\, dx'. \tag{A.12}$$

Intuitively, if the chain starts with $X_0 \sim \pi$, then $X_t \sim \pi$ for all $t \geq 0$. Under irreducibility and aperiodicity, the distribution p_t converges to π as $t \to \infty$ regardless of the starting distribution:

$$\lim_{t\to\infty} p_t(x) = \pi(x). \tag{A.13}$$

Irreducibility. A Markov chain with transition kernel $q(x|x')$ on state space $\mathcal{X}$ is called *irreducible* if every state can be reached from every other state in a finite number of steps with positive probability. Formally, for any two states $x, x' \in \mathcal{X}$ there exists an integer $n \geq 1$ such that

$$q^{(n)}(x|x') > 0, \tag{A.14}$$

where $q^{(n)}$ is the n-step transition probability. Irreducibility ensures that the chain does not split into isolated subsystems and that probability mass can flow across the entire state space.

Aperiodicity. A state x has period $d \geq 1$ if d is the greatest common divisor of all integers $n \geq 1$ for which the chain can return to x in exactly n steps with positive probability:

$$d = \gcd\{n \geq 1 : q^{(n)}(x|x) > 0\}. \tag{A.15}$$

If $d = 1$, the state is said to be aperiodic. A Markov chain is aperiodic if all of its states are aperiodic. Aperiodicity prevents the chain from getting locked into deterministic cycles of fixed length.

A.3. Transformers

Transformers, based on the *multi-head attention mechanism* introduced in 2017 by Vaswani et al. [97] gained popularity due to their use in Natural Language Processing (NLP).

Transformers work by converting the input text into discrete numerical representations, designated by *tokens*, which are then mapped to dense vector representations through a learned embedding lookup table that maps tokens to dense vectors.

Since Transformers lack inherent sequence awareness, *positional encodings* such as sine/cosine functions are added to token embeddings to inject order information. At each layer of the network, a *multi-head attention mechanism* dynamically *weighs relationships between tokens* within a defined context window, allowing the model to amplify signals from relevant tokens, while suppressing irrelevant ones. This self attention process progressively builds *contextual understanding*, enabling transformers to capture long range dependencies and hierarchical patters in the data.

A.3.1 Multi-head attention mechanism

Firstly, three vectors are derived through linear transformations of the token- a *Query (Q)*, representing *"what the token is looking for"*, a *Key (K)*, representing *"what the token can offer"*, and a *Value (V)*, storing the *actual information* that is going to be used to update the representation of the query.

Let d_k be the dimension of the key/query vectors. The *attention scores* are *pairwise similarity score*s between queries and keys across all tokens in the context window (Equation A.16).

$$\text{Attention}(Q, K, V) = \text{softmax}\Big(\frac{QK^T}{\sqrt{d_k}}\Big)V \tag{A.16}$$

The self-attention process is repeated in parallel across *multiple "heads"*, each learning different interaction patterns (syntactic, semantic, positional relationships, etc.). Then, the *outputs from all of the heads are concatenated and linearly projected* in order to form the final output, which are tokens that dynamically attend to relevant parts of the input.

After attention, each token's representation is passed through a *position-wise FFN*. This network consists of *two linear layers with a ReLU/GELU activation* in between, and refines each token's representation independently, adding non-linearity to the model.

A.4. Vision Transformers (ViT)

In 2019, Ramachandran et al. [98] adapted the Transformers architecture to Computer Vision tasks. They propose a *self-attention-based architecture that replaced convolutional operations* and demonstrate that purely attention-based mechanisms have the potential to rival convolutional neural networks (CNNs) in tasks like image classification, paving the way for *Vision Transformers (ViT)*.

This was achieved by treating images as sequences of flattened 2D patches, designated *visual tokens*, which were then processed through *scaled dot-product attention* to model long-range spatial dependencies, instead of CNNs' local receptive fields.

Vision Transformers (ViTs), introduced in 2021 by Dosovitskiy et al. [99], are transformers designed for Computer Vision in alternative to Convolutional Neural Networks (CNNs). Given an input image, a ViT breaks it down into several *disjoint patches*, converts each patch into a vector, and maps the patches into a lower-dimensional feature space through a single matrix multiplication (linear projection layer). The resulting embeddings are then passed through a transformer embedding, as if they were token embeddings.

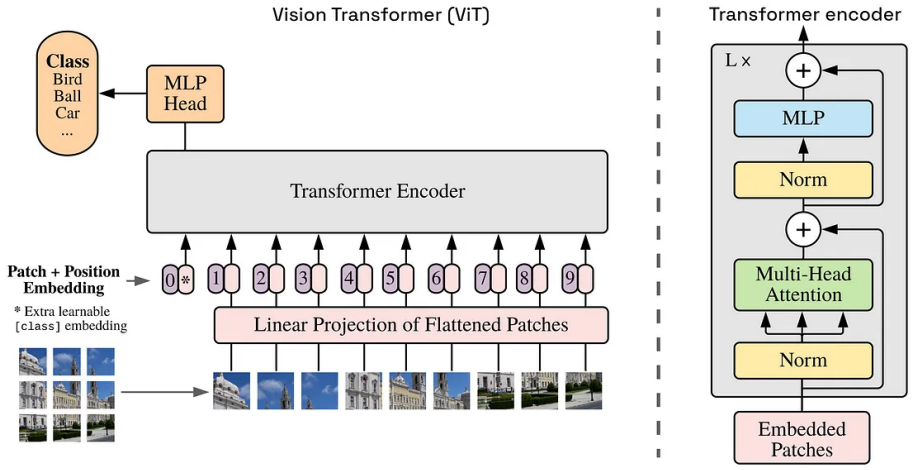


Figure A.3: Illustration from the original ViT paper [99]. An input image is split into fixed-size patches, which are linearly projected into patch embeddings and combined with positional encodings. These embeddings are processed by a standard Transformer encoder using multi-head self-attention, producing a representation suitable for classification or other vision tasks. Unlike CNNs, ViTs rely entirely on attention mechanisms to model spatial relationships between patches.

A.5. Vector-Quantized Generative Adversarial Network (VQ-GAN

Vector-Quantized Generative Adversarial Networks (VQ-GANs), proposed by Esser et al. [76], are generative models that learn a discrete latent representation of images and use adversarial training to produce high-fidelity reconstructions.

In a VQ-GAN an encoder E maps an input image $x \in \mathbb{R}^{H\times W\times 3}$ into a low-dimensional feature map $z_e \in \mathbb{R}^{h\times w\times d}$. Each spatial location of z_e is then quantized to the nearest entry of a learned codebook $\mathcal{Z} = \{e_k\}_{k=1}^{K}$, producing a discrete latent representation z_q. A decoder D reconstructs the image $\hat{x} = D(z_q)$ from the quantized latents.

Compared to VQ-VAE, VQ-GAN adds a patch-based discriminator that distinguishes between real and reconstructed image patches, and its gradients encourage the decoder to produce sharper, more perceptually realistic outputs. A perceptual loss based on a pre-trained feature extractor (e.g. VGG) is also commonly included. The resulting objective combines a reconstruction term, a codebook commitment term, an adversarial loss and optionally a perceptual loss:

$$\mathcal{L} = \|x - \hat{x}\|_1 + \beta\|\mathrm{sg}[z_e] - e\|_2^2 + \gamma\|z_e - \mathrm{sg}[e]\|_2^2 + \lambda\mathcal{L}_{adv} + \eta\mathcal{L}_{percep},$$

where $\mathrm{sg}[\cdot]$ denotes the stop-gradient operator and the coefficients control the relative importance of each component.

By learning a discrete, spatially structured latent space with adversarially trained reconstructions, VQ-GAN achieves compression, semantic organization (VQ), sharpness and fidelity (GANs). This makes it an attractive building block for transformer or diffusion priors, which can then operate in the compressed latent domain rather than directly in pixel space.

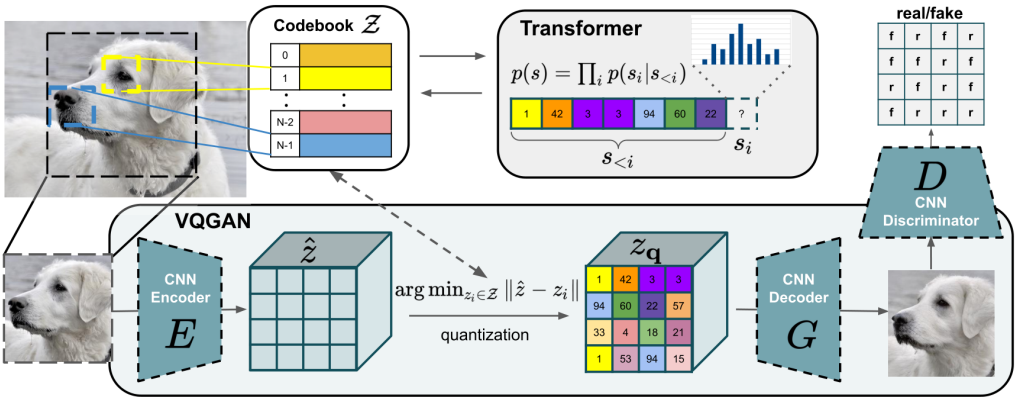


Figure A.4: Scheme of a VQ-GAN. From the original paper [76].

B. Methodology

B.1. Training

B.1.1 Hyper-parameters

Table B.1: Default training hyper-parameters used for IDiff-Face (baseline and ethnicity-specific).

Parameter	**Value**
Latent diffusion variant	`latent_diffusion = true`
Noise schedule	DDPM
Backbone	Cross-attention U-Net
Context dropout (CPD)	25 % (`context_dropout = 0.25`)
Context permutation	None (`context_permutation = 0.0`)
Seed	Fixed (unless stated otherwise)
Precision	Mixed (“16-mixed”)
Batch size	128
Data-loading workers	4 (pinned memory)
Training steps	150k
Optimizer	Adam
Learning rate	2×10^{-4}
Scheduler	Cosine-annealing warm-restarts ($T_0 = 10{,}000$, $T_{\text{mult}} = 2$)
EMA of weights	inv_gamma=1.0, power=0.75, min=0.0, max=0.9999

B.1.2 Validation during training

We generated a horizontally concatenated grid with four blocks, as follows:

1. First block: images generated by sampling from the model without any conditioning vector.
2. Second block: images generated by sampling conditioned on a randomly generated and normalized noise vector.
3. Third block: images were generated using a single synthetic context vector.
4. Fourth block: real images.

Baseline single-model IDiff-Face Figure B.1 illustrates the sample quality during training of the IDiff-Face model without ethnicity conditioning at step $150K$.

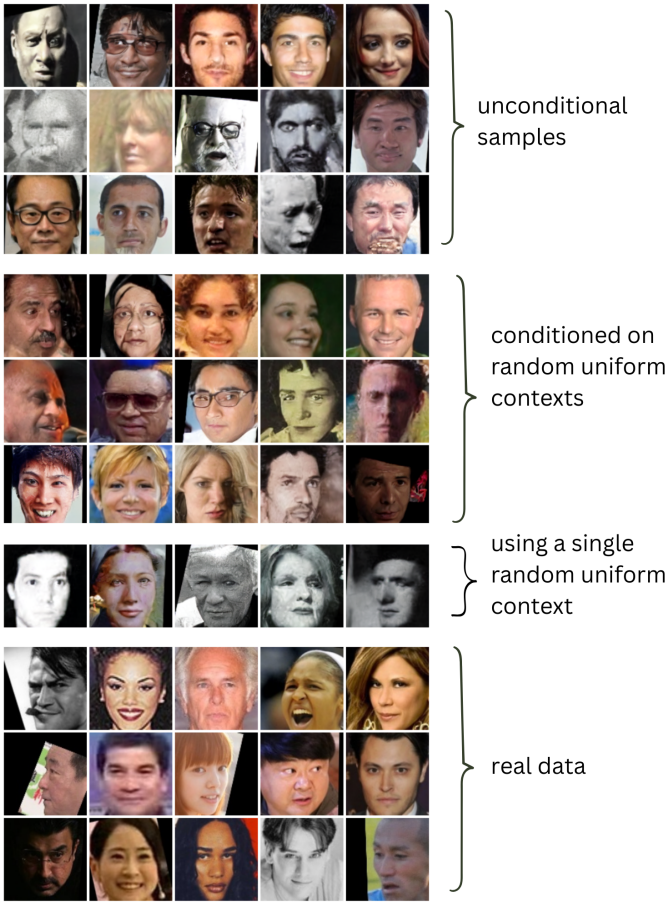


Figure B.1: Grid of synthetic face samples generated during training of the baseline IDiff-Face model on the full BalancedFace dataset.

Our ethnicity-specific approach Below, in Fig. B.2, we can see the grid of samples generated while training the ethnicity-specific IDiff-Face models, at step 150K.



Figure B.2: Sampling during training at denoising step 150K using our ethnicity specific approach.

B.2. FR Model Training and Evaluation

This appendix lists the hyper-parameters and settings used when evaluating the utility of the generated images for training face recognition (FR) models.

XNorm corresponds to the average L2 norm of the embeddings produced by the FR model. Prior work [20] has shown that the embedding norm is correlated with the discriminative power of the features and can be used as an indicator of confidence in face verification tasks. Although XNorm is not itself a performance metric, it serves as a diagnostic measure to highlight potential disparities in representation quality across groups.

Table B.2: Hyper-parameters and settings used to train and evaluate the ResNet-18 face recognition model with ArcFace loss.

Setting	**Value**
Backbone	ResNet-18
Loss	ArcFace (scale $s = 64.0$, angular margin $m = 0.35$)
Dropout (backbone)	$p = 0.4$
Optimizer	SGD
Learning rate	0.1 (decayed $\times 0.1$ at scheduled epochs)
Momentum	0.9
Weight decay	5×10^{-4}
Max epochs	200
Batch size	128
Data loading workers	8
Early checkpointing	stop when validation average no longer improves (training loop implementation)
Input preprocessing	per-channel normalization to $[-1, 1]$
Data augmentation	RandAugment with random horizontal flip + moderate geometric/photometric transforms
Test-time augmentation	horizontal flip; average embeddings prior to L2 normalization (“Accuracy (Flip)”)
Verification protocol	RFW test subset, four ethnicity groups (African, Asian, Indian, Caucasian), 10-fold protocol
Threshold selection	per fold, chosen on training split; report accuracy on held-out split
Reported metrics	mean accuracy $\pm$ std over 10 folds; TAR@FAR=10^{-3} (mean $\pm$ std); XNorm (average L2 norm of embeddings)

C. Results: Classifier for the contexts

The confusion matrix in Fig C.1 presents the high values along the diagonal, confirming our classifier's strong performance across all four ethnicity groups, with minimal misclassification between classes.

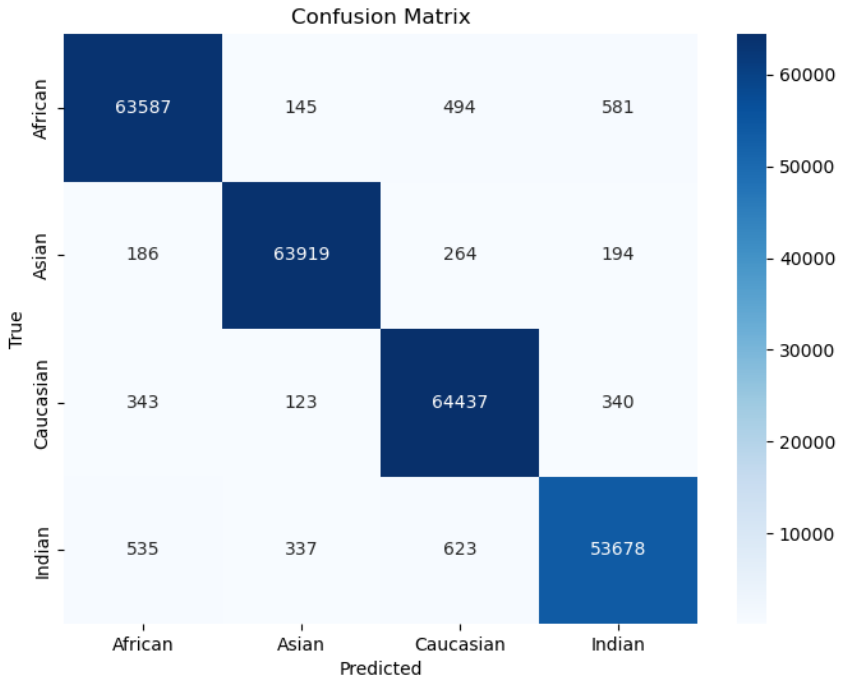


Figure C.1: Confusion matrix of the ethnicity classifier evaluated on a validation set taken from the BalancedFace dataset. Each cell represents the number of predictions for a given true–predicted class pair.

In Fig C.2, we present the training and validation losses of the ethnicity classifier, showing that the model converges stably. We note that the validation loss is slightly lower than the training loss. This can occur naturally when the training loss is computed with stochastic regularization (dropout at rate 0.3) and weight penalties active, whereas the validation loss is computed with the model in inference mode without dropout or regularization. This difference does not necessarily indicate data leakage or overfitting.

In Fig C.3, we present the training and validation accuracies of the classifier, which demonstrate strong generalization capability and consistent performance on the validation set.

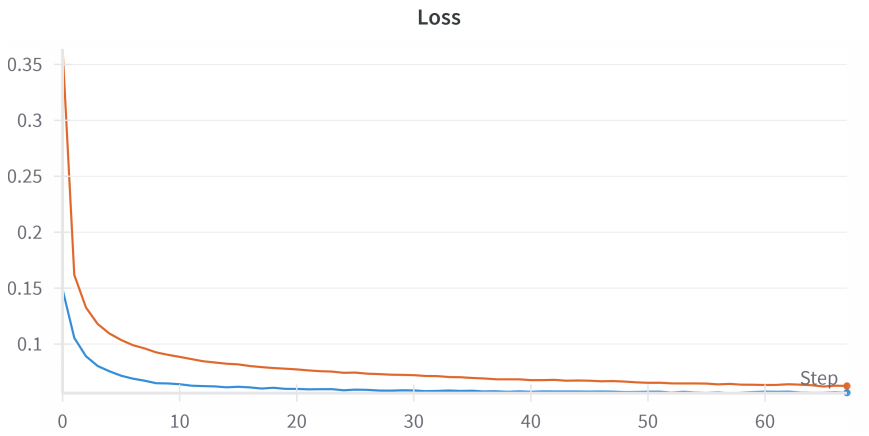


Figure C.2: Training (orange) and validation (blue) losses.

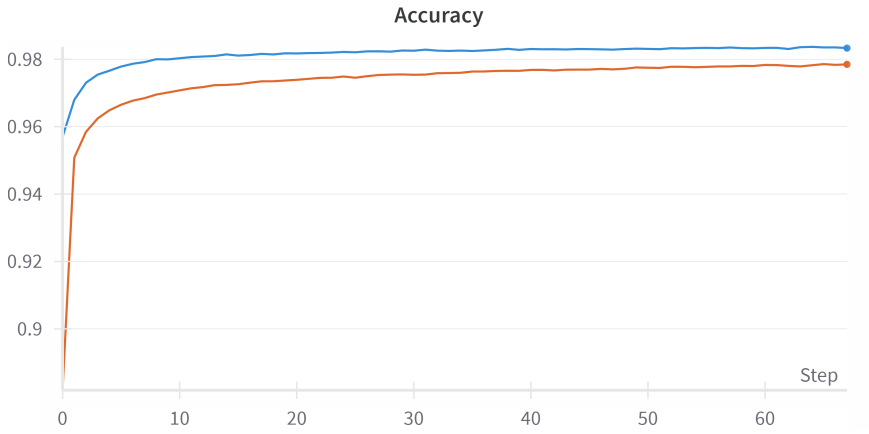


Figure C.3: Training (orange) and validation (blue) accuracies.