INSTITUT FÜR
INFORMATIK
Datenbanken und Informationssysteme

Universitätsstr. 1 D–40225 Düsseldorf

Heinrich Heine
Universität
Düsseldorf

Debiasing Text-to-Image Diffusion Models

Maximilian Bertram

Master Thesis

Date of issue:	08. April 2024
Date of submission:	07. October 2024
Reviewers:	Prof. Dr. Stefan Conrad
	Prof. Dr. Martin Mauve

Erklärung

Hiermit versichere ich, dass ich diese Master Thesis selbstständig verfasst habe. Ich habe dazu keine anderen als die angegebenen Quellen und Hilfsmittel verwendet.

Düsseldorf, den 08.Oktober

Maximilian Bertram

Abstract

Text-to-Image (TTI) diffusion models have demonstrated remarkable capabilities in generating high-quality images from textual descriptions. However, these models often inherit and amplify societal biases present in their training data, particularly concerning sensitive attributes such as gender, race, and age. This thesis introduces Debias Diffusion, a novel approach to mitigate such biases in TTI diffusion models during the inference stage. Debias Diffusion operates by dynamically modifying the generative process, leveraging the model's inherent semantic understanding and latent space representations. The method employs syntactic analysis of input prompts and lightweight classifiers operating in the model's h-space, a rich semantic latent space spanned by the outputs of the UNet encoder. By inserting carefully chosen attribute modifiers into prompts at dynamically determined timepoints, the algorithm guides the generation towards fairer outputs while preserving original image semantics and quality. We conducted extensive empirical evaluations, generating and analyzing over 1.5 million images across various prompting scenarios. Results demonstrate that Debias Diffusion consistently achieves superior fairness compared to the base model and competitive state-of-the-art methods. The approach shows particular strength in simultaneously debiasing multiple attributes while maintaining high image quality and semantic coherence. Another key contribution of this work is a new, flexible method for training h-space classifiers. This self-labeled approach generates a balanced dataset using the model itself, guided by carefully crafted prompts. It eliminates the need for external labeled datasets, making the method more adaptable and reducing dependencies. Our method addresses key challenges in debiasing generative models: complexity, performance, and usability. Debias Diffusion introduces minimal computational overhead, allows for flexible adaptation to different target distributions, and operates without requiring expensive model retraining or finetuning. This work contributes to the ongoing efforts to create more equitable and responsible AI systems, offering a practical solution for reducing harmful biases in widely-used generative models while preserving their utility and performance. The complete implementation of Debias Diffusion as well as additional scripts to recreate all presented results are available at https://github.com/MaxBer04/DebiasDiffusion.

Contents

1 Introduction

Machine learning has long aimed to develop efficient and expressive generative models capable of capturing real-world processes from observed data. This goal has recently become attainable, even for high-dimensional data, through advanced probabilistic models. Diffusion models (Sohl-Dickstein et al., 2015; Y. Song and Ermon, 2019; Ho et al., 2020), which form the focus of this work, have shown particular promise in this regard. These frameworks combine techniques from Bayesian statistics and information theory to provide accurate probabilistic approximations of real data distributions. The adoption of diffusion models has accelerated with the development of efficient guidance methods (Ho and Salimans, 2021) and the release of high-performance, open-source models that can process free-form text input (Rombach et al., 2021; Podell et al., 2024; Esser et al., 2024). This progress has been complemented by advances in language modeling and multimodal domains (Devlin et al., 2019; Radford et al., 2021).

1.1 Problem Statement

The widespread use of these models has raised concerns regarding fairness in generative modeling, reinforced by known biases in many publicly available training datasets, particularly those related to human attributes such as gender, race, or age (Kärkkäinen and Joo, 2021). As discussed in Section 3.1, current diffusion models often inherit or amplify these biases. Recent research has focused on identifying sources of such biases and developing mitigation strategies, as outlined in Section 3.2. These approaches can be broadly categorized into dataset-centric and model-centric methods, with the latter including all auxiliary components of the generative pipeline and posing the focus of this work.

1.2 Thesis Structure and Contribution

This thesis explores these challenges and potential solutions with an equal focus on syntactic and semantic aspects. Section 2 provides the necessary theoretical background, while Section 3 examines fairness in generative modeling, contextualizing emergent biases and reviewing existing model-based debiasing approaches. The main contributions of this work are twofold: it presents a novel debiasing method (Section 4) and offers a comprehensive empirical evaluation of this method alongside three state-of-the-art debiasing techniques for diffusion models comprising over 1.5 million generated images (Section 5). As part of the developed method, this thesis analyzes a semantic latent space used by most current diffusion models, initially identified by Kwon et al. (2023), and introduces a new flexible training method for lightweight classifiers operating in this latent space with minimal external dependencies. Section 6 provides ablation studies on key hyperparameters of the proposed method, revealing interesting properties of the aforementioned classifiers. The thesis concludes in Section 7 with a summary of the findings and suggestions for future research in the area of debiaising diffusion models.

2 Theoretical Background

This section provides the theoretical foundation for generative modeling of data using unsupervised machine learning techniques. The discussion begins with an introduction to maximum-likelihood estimation, followed by an exploration of deep latent variable models, and concludes with a detailed examination of diffusion models. Consider a dataset

$$\mathcal{D} := \{\boldsymbol{x}^n\}_{n=1}^N \subset \mathcal{X}^N$$

where each $\boldsymbol{x}^n$ represents a real-world observation or simulation captured in some manifold $\mathcal{X} \subset \mathbb{R}^{d_1 \times \ldots \times d_M}$. For improved readability, the abbreviated notation $\boldsymbol{x}^{[1:N]}$ is primarily used. Throughout this work, vectors and matrices are denoted using bold notation. The goal of generative modeling is to generate new "real" datapoints that capture the underlying patterns of the observed data. It is commonly assumed that these samples are obtained from an unknown, stationary process. Modeling this process often requires uncovering hidden relationships between observed data, referred to as unobservable or *latent variables* (Heaton, 2017, §3.9.6). Formally, we assume a relationship:

$$\boldsymbol{x} = f(\boldsymbol{z}, \boldsymbol{\epsilon}) \tag{1}$$

where $f(\cdot, \cdot)$ is a function that defines the resulting data given a latent variable $\boldsymbol{z} \in \mathcal{Z} \subseteq \mathcal{X}$ and a random noise component $\boldsymbol{\epsilon}$. This formulation allows f to be either deterministic (when $\boldsymbol{\epsilon}$ is absent or fixed) or stochastic (when $\boldsymbol{\epsilon}$ is a random variable). In this context, $\boldsymbol{z}$ represents the underlying latent structure or signal inherent in the observations, while $\boldsymbol{\epsilon}$ captures the inherent randomness or noise present within them.

2.1 Maximum-Likelihood Estimation

While Equation (1) provides a conceptual framework, its direct modeling is often impractical. Instead, the process is commonly assumed to be random with a *probability density function (PDF)* $p(X = \boldsymbol{x})$ and a continuous random variable X. When the random variable is clear from context, we abbreviate using $p(X = \boldsymbol{x}) = p(\boldsymbol{x})$. Similarly, for a discrete random variable X, we use $P(X = \boldsymbol{x}) = P(\boldsymbol{x})$ to denote the corresponding *probability mass function (PMF)*. Both are referred to simply as (probability) distributions throughout this work (Heaton, 2017, §3.2). Most commonly used distributions can be defined entirely by a fixed set of parameters, e.g., mean and variance $\boldsymbol{\theta} = (\boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$ of a (multivariate) *Gaussian*, formally given by

$$\mathcal{N}\left(\boldsymbol{x}; \boldsymbol{\mu}, \sigma^2 \boldsymbol{I}\right) \widehat{=} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{\boldsymbol{x} - \boldsymbol{\mu}}{\sigma}\right)^2\right) \tag{2}$$

where $\boldsymbol{I}$ denotes the identity matrix. Note that throughout this work, all multiplications of scalars and matrices are defined to be element-wise unless specified differently. The distribution in Equation (2) is also commonly known as a (multivariate) *normal distribution*. Generally, a parametrized PDF is denoted as $p_{\boldsymbol{\theta}}(\boldsymbol{x}) := p(\boldsymbol{x}; \boldsymbol{\theta})$. Leveraging all observations, the most complete or holistic description of the underlying process is given by the joint distribution:

$$p_{\boldsymbol{\theta}}(\mathcal{D}) := p\left(\boldsymbol{x}^{[1:N]}\right) = p_{\boldsymbol{\theta}}\left(\boldsymbol{x}^1, \ldots, \boldsymbol{x}^N\right) \tag{3}$$

over all observed variables (Heaton, 2017, §3.3).

The conceptual approach to selecting suitable parameters for this distribution is as follows: The aim is to find a parametrization $\boldsymbol{\theta} \in \mathcal{P}$ in the space of possible choices that assigns very high probabilities to observed datapoints as opposed to random ones. If we can assume that past observations indeed allow for useful future predictions, i.e., if the method generalizes in a suitable manner and does not simply memorize data, sampling from this distribution should yield new data that could have been part of the observed data with high likelihood. Typically, this objective is formulated as:

$$\boldsymbol{\theta}_{MLE} = \arg\max_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{[1:N]} \right) = \arg\max_{\boldsymbol{\theta}} \log \prod_{n=1}^{N} p_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{n} \right) = \arg\max_{\boldsymbol{\theta}} \sum_{n=1}^{N} \log p_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{n} \right) \quad (4)$$

where the i.i.d. assumption of the data is leveraged to factorize the probabilities, and the log-likelihood is used for better numerical stability. Objective (4) is termed the *maximum likelihood estimation (MLE)* with respect to the parameters $\boldsymbol{\theta}_{MLE} \in \mathcal{P}$ (Heaton, 2017, §5.5).

The implicit assumption of this approach is that the dataset itself is holistically modeled by some unknown, true probability distribution $p(X = \boldsymbol{x})$. For sufficiently large sample sizes N, the Cramér-Rao lower bound proves the optimality of Objective (4) in an L2-sense such that:

$$\| \boldsymbol{\theta}_{MLE} - \boldsymbol{\theta}^{*} \|_2^2 \leq \left\| \hat{\boldsymbol{\theta}} - \boldsymbol{\theta}^{*} \right\|_2^2 \quad (5)$$

for any alternative parametrization $\hat{\boldsymbol{\theta}} \in \mathcal{P}$ and optimal solution $\boldsymbol{\theta}^{*} \in \mathcal{P}$ (Heaton, 2017, §8.3.1). Therefore, optimization using the MLE objective is expected to yield optimal parameters $\boldsymbol{\theta}^{*}$ with $p_{\boldsymbol{\theta}^{*}}(\boldsymbol{x}) = p(\boldsymbol{x})$. Moreover, for large datasets with $N \to \infty$, MLE is provably the fastest statistical method to obtain such optimal parameters $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}^{*}$ (Ermon, 2023). Due to its relevance for this work, we will focus solely on such likelihood-based approaches to generative modeling. Intuitively, optimizing the MLE criterion is equivalent to obtaining better lossless compressions of the data (Ermon, 2023). However, it should be noted that it has been explicitly shown that sample quality and data likelihoods are largely uncorrelated for high-dimensional data. For example, optimizing models with the MLE objective and datasets assumed to contain a lot of noise or noisy samples ($\boldsymbol{\epsilon} \gg \boldsymbol{z}$ in Equation (1)) yields generative results of inferior quality (Theis et al., 2016). A prominent contemporary example of a generative architecture for high-dimensional data independent of likelihood estimations is the *Generative Adversarial Network (GAN)* (Goodfellow et al., 2014).

A very recent and successful line of research has employed techniques from the domain of machine learning to find optimal parametrizations. As such, a suitable learning signal is required that serves as a proxy to the overall goal. This is directly inferred from Objective (4) as

$$\mathcal{L}(\boldsymbol{\theta}) = \sum_{n=1}^{N} \log p_{\boldsymbol{\theta}} \left(\boldsymbol{x}^{n} \right) \quad (6)$$

where $\mathcal{L}(\cdot)$ is called the *loss function*. Optimization is typically performed using gradient-based methods to backpropagate the loss from Equation (6) to the parameters $\boldsymbol{\theta}$, formally denoted as $\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x})$. This approach has shown to perform especially well for high-dimensional data which is mostly attributed to a lower probability of converging in local

minima of $\mathcal{L}$. It is commonly termed *batch gradient descent* and scales linearly with dataset size N, which can be too expensive to compute for many use-cases (Heaton, 2017, §8.1.3). A common solution is to use multiple random minibatches $\mathcal{M} \subset \mathcal{D}$ of size $|\mathcal{M}| = M$ as an unbiased estimator to compute stochastic gradients:

$$\frac{1}{N}\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathcal{D}) \simeq \frac{1}{M}\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\mathcal{M}) = \frac{1}{M}\sum_{\boldsymbol{x}\in\mathcal{M}} \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \tag{7}$$

This method is called *stochastic gradient descent (SGD)* with $\simeq$ denoting that both sides are equal when averaged over the noise introduced by random selection of mini-batches (Heaton, 2017, §5.9). This is typically accounted for by iterating over the whole dataset multiple times, denoted as *epochs*. The method can be further improved upon by plugging the resulting gradients into stochastic gradient-based optimizers such as ADAM (Diederik P. Kingma and Ba, 2014).

2.2 Deep Latent Variable Models

To compute the loss in Equation (6), access to the marginal distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ is required. This necessitates a return to the latent variable formulation in Equation (1). A common approach to address the intractability of unobservable latents involves the following factorization of the joint distribution:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z}) \tag{8}$$

In this formulation, $p(\boldsymbol{z})$ represents a distribution over latents chosen according to domain knowledge present prior to observing any data, hence referred to as the *prior distribution* (Heaton, 2017, §3.9.6). A typical choice in uncertain scenarios is a simple Gaussian (see Equation (2)). Their prevalence in this context can partially be attributed to their relation to the central limit theorem (Heaton, 2017, §3.9.3). Furthermore, the Gaussian distribution maximizes entropy among continuous distributions with specified variance, making it a symbol of uncertainty rather than knowledge (Heaton, 2017, §19.4.1). The marginal distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ would consequently be obtained as:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} = \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z}) d\boldsymbol{z} \tag{9}$$

by marginalizing over latents. This is also denoted as the *marginal likelihood* or *model evidence* with respect to the parametrization $\boldsymbol{\theta}$ (Heaton, 2017, §3.9.4). A simple example would involve a discrete latent $Z \sim \text{Categorical}(1, \ldots, K)$ chosen from a categorical distribution over K classes and a Gaussian posterior $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid Z = k) = \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ with parameters $\boldsymbol{\theta} \coloneqq (\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ depending on the conditioned class. The model evidence could then be computed as:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int p_{\boldsymbol{\theta}}(\boldsymbol{x}, Z) dZ = \sum_{k=1}^{K} \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) p(Z = k). \tag{10}$$

with a chosen categorical prior $p(Z = k)$. Such a compound distribution $p_{\boldsymbol{x}}(\boldsymbol{x})$, termed a *mixture-of-Gaussians* in this case, can be quite powerful and expressive. For continuous

latents, which are the focus of this work, $p_{\boldsymbol{\theta}}(\boldsymbol{x})$ would be a potentially more powerful, infinite mixture (Heaton, 2017, §3.9.6).

In practice, the integral in Equation (9) is often too complex or expensive to compute. For illustration, consider the previous example with a latent $Z \in \{0,1\}^{20}$ encoding 20 features. Evaluating Equation (10) would involve computation of over 2^{20} terms. Generally, marginalizing over all possible latents is too unstable and complex. To address this, it would be beneficial to have a probability distribution $q(\boldsymbol{z}, \boldsymbol{x})$ at hand that yields samples encoding much of the variance observed in the data. This would enable a much more efficient marginalization:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \int_{\mathcal{Z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} = \int_{\mathcal{Z}} \frac{q(\boldsymbol{z} \mid \boldsymbol{x})}{q(\boldsymbol{z} \mid \boldsymbol{x})} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) d\boldsymbol{z} = \mathbb{E}_{\boldsymbol{z} \sim q(\boldsymbol{z}|\boldsymbol{x})} \left[\frac{p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x})}{q(\boldsymbol{z} \mid \boldsymbol{x})} d\boldsymbol{z} \right] \tag{11}$$

using Monte-Carlo estimates with 'relevant' samples $\boldsymbol{z} \sim q(\boldsymbol{z} \mid \boldsymbol{x})$ instead of samples $\boldsymbol{z} \in \mathcal{Z}$ over the whole domain. Theoretically, this is motivated by the following alternative factorization of the joint distribution compared to Equation (8):

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x}) p_{\boldsymbol{\theta}}(\boldsymbol{x}) \;\Leftrightarrow\; p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x})}. \tag{12}$$

Therefore, assuming a posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \approx p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x})$ using a new set of parameters $\boldsymbol{\phi}$, the expectation in Equation (11) can be computed accurately and efficiently. Due to their relations, $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ is also referred to as the *encoder model* and $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})$ as the *decoder model* (D. Kingma, 2017, §2.2).

Feed-forward differentiable neural networks have proven to be powerful function approximators for distributional parameters, suitable for optimization with SGD (see Equation (7)). Of particular prominence are deep neural networks, which employ multiple layers of hidden units. Models with latent variables such as those described in Equation (8) that utilize deep neural networks are termed *deep latent variable models (DLVMs)* (D. Kingma, 2017, §1.6). For instance, the encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ typically employs neural network architectures designed to encode meaningful latents through compression by using a smaller latent space than data space. Formally, this is denoted as:

$$\begin{aligned} (\boldsymbol{\mu}, \log \boldsymbol{\sigma}) &= \text{EncoderNeuralNet}_{\boldsymbol{\phi}}(\boldsymbol{x}) \\ q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) &= \mathcal{N}(\boldsymbol{z}; \boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma})) \end{aligned} \tag{13}$$

with $\boldsymbol{z} \in \mathcal{Z} \subset \mathcal{X}$. Correspondingly, the decoder is designed to cast compressed latents back into the data space. In these setups, which form the focus of this work, the parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ denote the weights and biases defining the modeled function approximation (or neural network) used to yield distributional parameters (D. Kingma, 2017, §2.2).

2.3 Variational Autoencoders and the ELBO

The objective of enabling stable computation of Equation (11) can be approached by reformulating it using the employed encoder and decoder models, along with the logarithm

for numerical stability:

$$\begin{aligned}
\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \mathbb{E}_q \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x})\right] \\
&\overset{(12)}{=} \mathbb{E}_q \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] \\
&= \mathbb{E}_q \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}{p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x}) q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] \\
&= \mathbb{E}_q \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] + \mathbb{E}_q \left[\log \left(\frac{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}{p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] \\
&= \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) + \mathrm{D}_{\mathrm{KL}} \left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{z} \mid \boldsymbol{x})\right)
\end{aligned} \tag{14}$$

where $\mathbb{E}_q \coloneqq \mathbb{E}_{\boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}$ for improved readability. The first equality holds as the evidence is independent of the latent variables. The second term in the last line, D_{KL}, denotes the *Kullback-Leibler divergence* between the true and approximate posteriors, which is always non-negative and zero only if the approximate posterior exactly equals the true posterior (Heaton, 2017, §3.13). Consequently, $\log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x})$, indicating that $\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x})$ provides a lower bound on the evidence, hence termed the *evidence lower bound (ELBO)*. This objective is also referred to as the *variational lower bound (VLB)*, as it involves varying over a family of distributions $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ using *variational parameters* $\boldsymbol{\phi}$ to optimize the lower bound (D. Kingma, 2017, §2.3).

The goal is to maximize the ELBO $\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}$, yielding a model that best approximates the evidence. For further insight, the ELBO can be reformulated using the KL-divergence:

$$\begin{aligned}
\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) &= \mathbb{E}_q \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] \overset{(8)}{=} \mathbb{E}_q \left[\log \left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] \\
&= \mathbb{E}_q \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right] + \mathbb{E}_q \left[\log \left(\frac{p(\boldsymbol{z})}{q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})}\right)\right] \\
&= \underbrace{\mathbb{E}_q \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right]}_{\text{reconstruction term}} - \underbrace{\mathrm{D}_{\mathrm{KL}} \left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z})\right)}_{\text{prior matching term}}
\end{aligned} \tag{15}$$

This formulation reveals two competing objectives: maximizing the reconstruction term and minimizing the prior matching term. The reconstruction term is maximized when the approximate posterior $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ encodes latents that allow for reconstructed data points with high likelihoods using the true posterior $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})$. The prior matching term can be interpreted as a regularization term ensuring that the variational encoder does not differ too greatly from the chosen prior over latents $p(\boldsymbol{z})$, which will be important for sampling purposes. The loss function $\mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x})$, together with an encoder $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ and decoder model $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})$, parametrized using neural networks similar to Equation (13), constitutes a *Variational Autoencoder (VAE)* (Diederik P. Kingma and Welling, 2014b; Rezende et al., 2014). In contrast to classical Autoencoders, which are trained by encoding and decoding datapoints using only the reconstruction loss as a learning signal, the VAE framework additionally optimizes the encoding space through the prior matching term. The overall

training objective can be expressed as:

$$\begin{aligned}\arg\max_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathcal{L}_{\boldsymbol{\theta},\boldsymbol{\phi}}(\boldsymbol{x}) &= \arg\max_{\boldsymbol{\theta},\boldsymbol{\phi}} \mathbb{E}_q \left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right] - \mathrm{D}_{\mathrm{KL}} \left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z})\right) \\ &\approx \arg\max_{\boldsymbol{\theta},\boldsymbol{\phi}} \frac{1}{L} \sum_{l=1}^{L} q_{\boldsymbol{\phi}}\left(\boldsymbol{z}^l \mid \boldsymbol{x}\right) \log p_{\boldsymbol{\theta}}\left(\boldsymbol{x} \mid \boldsymbol{z}^l\right) - \mathrm{D}_{\mathrm{KL}} \left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p(\boldsymbol{z})\right)\end{aligned} \tag{16}$$

where the reconstruction term is approximated as a Monte Carlo estimate over a finite set of latents $\boldsymbol{z}^{[1:L]}$ sampled from $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ for each datapoint $\boldsymbol{x}$. However, each $\boldsymbol{z}^l$ is obtained through a stochastic sampling procedure, which is generally non-differentiable, rendering this objective intractable for gradient-based optimization with respect to $\boldsymbol{\phi}$. This challenge can be addressed using a reparametrization trick:

$$\boldsymbol{z}^l = \mu_{\boldsymbol{\phi}}(\boldsymbol{x}) + \sigma_{\boldsymbol{\phi}}(\boldsymbol{x})\boldsymbol{\epsilon} \quad \text{with} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{17}$$

This trick, rediscovered from general probability theory as a beneficial property for differentiating DLVMs, was independently proposed by three research groups around the same time (Diederik P. Kingma and Welling, 2014a; Rezende et al., 2014; Titsias and Lázaro-Gredilla, 2014). It leverages the fact that arbitrary Gaussians can be represented as a standard Gaussian with accordingly shifted mean $\mu_{\boldsymbol{\phi}}(\boldsymbol{x})$ and stretched variance $\sigma_{\boldsymbol{\phi}}(\boldsymbol{x})$. In this setup, $\boldsymbol{z}$ becomes a deterministic function of $\boldsymbol{x}$ and $\boldsymbol{\epsilon}$ a stochastically sampled noise variable. This approach permits the use of Monte-Carlo estimates with reasonable variances to compute gradients with respect to $\boldsymbol{\phi}$, consequently allowing for joint optimization of the parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ of a VAE using the ELBO objective (16). It is important to note that this reparametrization is generally only possible for continuously distributed latents (such as Gaussians). This limitation arises from the fact that the reparametrization trick inherently avoids computation of gradients of an expectation by computing expectations of gradients, which are likely not defined for discrete latents (Luo, 2022). DLVMs with discrete latent variables, often referred to as *belief networks*, require different measures to allow for gradient-based optimization of distributional parameters, such as score functions (see Section 2.4.2) or control variates (Mnih and Gregor, 2014) among others (Heaton, 2017, §19.4.1).

Once a VAE is trained, new data can be generated by sampling latents $\boldsymbol{z} \sim p(\boldsymbol{z})$ from the unconditional prior and decoding them using the learned model $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})$. It is worth noting that the encoder model $q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x})$ is not directly involved in the generative process. However, due to the prior matching term in Equation (15), sampling latents from $p(\boldsymbol{z})$ should yield representations similar to those obtained by encoding the original datapoints, and therefore suitable for decoding with $p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})$. The VAE framework has demonstrated impressive results in various complex domains, including image and music generation (Imran and Terzopoulos, 2019; T. Wang et al., 2020) as well as text modeling (R. Li et al., 2020). In many cases, the learned latent space is much lower-dimensional than the data space, as well as semantically interpretable and separable, enabling applications such as image inpainting (Tu and Chen, 2019).

2.4 Diffusion Models

A popular approach for enhancing the generative capabilities of VAEs involves recursively stacking multiple VAEs. This technique yields a set of latents $\boldsymbol{z}_{[1:T]}$ that can depend

on each other, enabling the modeling of more intricate, complex relationships within the data. For computational feasibility, the transitions are assumed to be hierarchical and Markovian, meaning each latent can only condition on the previous latent in a hierarchical chain. The joint and posterior distributions are then given as:

$$\begin{aligned} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}_{1:T}) &= p(\boldsymbol{z}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}_1) \prod_{t=2}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1} \mid \boldsymbol{z}_t) \\ q_{\boldsymbol{\phi}}(\boldsymbol{z}_{1:T} \mid \boldsymbol{x}) &= q_{\boldsymbol{\phi}}(\boldsymbol{z}_1 \mid \boldsymbol{x}) \prod_{t=2}^{T} q_{\boldsymbol{\phi}}(\boldsymbol{z}_t \mid \boldsymbol{z}_{t-1}) \end{aligned} \tag{18}$$

This framework is termed a *Markovian Hierarchical VAE (MHVAE)* (Luo, 2022). Typically, $p(\boldsymbol{z}_T)$ is chosen as a very simple distribution that can be sampled efficiently in closed form, possibly incorporating additional prior knowledge about the process or structure of latents. At the other end of the hierarchical chain is the modeled data distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x})$, which may be much more complex and expressive. As such, the HMVAE framework gradually converts samples between both distributions, an idea originally employed in the fields of statistical physics (Jarzynski, 1997) and sequential Monte-Carlo (Neal, 1998). This process essentially resembles the statistical concept of diffusion, analogous to the physical phenomenon where particles flow from regions of higher density to those of lower density. In this context, the particles would be data samples $\boldsymbol{x} \in \mathcal{D}$ which are gradually corrupted throughout the Markov chain as the overall distribution diffuses. The overarching idea is to learn how to revert this process in order to obtain data samples from noise.

In a seminal work by Sohl-Dickstein et al. (2015), a new variation of MHVAE architecture was introduced, allowing for a trade-off between flexibility and tractability suitable for modeling complex, high-dimensional PDFs $p(\boldsymbol{x})$. Due to its physical interpretation, it was termed *Diffusion Probabilistic Model (DPM)*. Originally, it was used to yield state-of-the-art estimations on data likelihoods for high-dimensional datasets, enabling applications such as outlier detection in image sets (Sohl-Dickstein et al., 2015). However, as previously noted, good likelihoods do not necessarily imply that the model can generate high-quality data samples, as was the case for the DPM framework (Ho et al., 2020). Since then, the diffusion model framework has been successfully modified to yield generative models capable of producing high-quality, realistic samples from high-dimensional datasets (e.g., images). In this regard, two largely isolated lines of research have emerged from the DPM framework, which are closely connected but provide different theoretical perspectives. These, along with other integral parts of contemporary, state-of-the-art diffusion model implementations, will be explained in the upcoming sections.

2.4.1 Denoising Diffusion Probabilistic Models

The diffusion model used in this work is based upon the modified framework developed by Ho et al. (2020), which is currently widely employed by many contemporary diffusion model implementations (Diederik Kingma and Gao, 2024; Rombach et al., 2021; Podell et al., 2024). The architecture, termed *Denoising Diffusion Probabilistic Model (DDPM)*, introduces three key restrictions to the MHVAE framework:

1. Latents and datapoints have the same dimensionality. Therefore, the notation

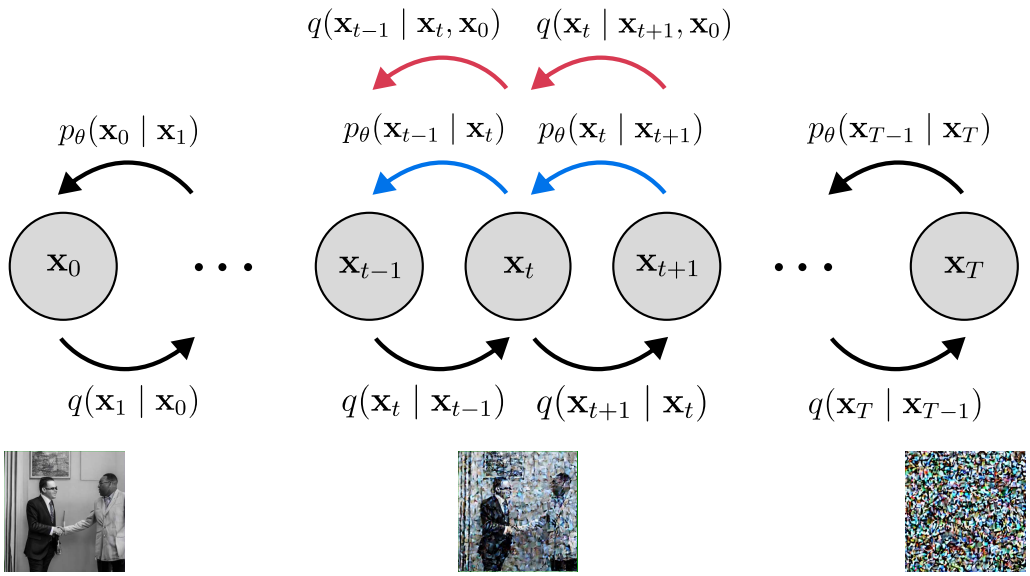


Figure 1: Illustration of an diffusion model as a special type of HMVAE. Blue arrows denote the backward process kernels which effectively learn the reverse forward kernels, denoted in red arrows. Images get gradually noisier as t increases. The figure is recreated from Luo (2022).

$\boldsymbol{x}_{[0:T]} \subset \mathcal{X}$ is used, where $\boldsymbol{x}_0 \in \mathcal{D}$ is a real data sample and $\boldsymbol{x}_{[1:T]}$ are hierarchical latents.

2. The encoders are modeled as fixed linear Gaussians such that:

$$\begin{aligned} q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right) &= \prod_{t=1}^{T} q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}\right) \text{ with} \\ q\left(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}\right) &= \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t \boldsymbol{I}\right) \end{aligned} \tag{19}$$

where $\beta_{[1:T]}$ is a fixed variance schedule with $1 \geq \beta_T \geq \ldots \geq \beta_1 \geq 0$ (see also Figure 2a). These transitions are denoted as the *forward process* using the prescribed, fixed Gaussian kernels $q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$.

3. The last latent in the chain is assumed to be a standard Gaussian devoid of any data signals. Hence, the learned *backward or generative process*, used to yield new data samples from Gaussian noise, is defined as:

$$\begin{aligned} p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{[0:T]}\right) &= p\left(\boldsymbol{x}_T\right) \prod_{t=1}^{T} p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t\right) \text{ with} \\ p\left(\boldsymbol{x}_T\right) &= \mathcal{N}\left(\boldsymbol{x}_T; \mathbf{0}, \boldsymbol{I}\right) \end{aligned} \tag{20}$$

The primary distinction between DDPMs and most other latent variable models lies in the fixed forward process. This process, schematically illustrated in Figure 1, represents a steady noisification of data until $\boldsymbol{x}_T \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ becomes pure noise. In the original formulation, Ho et al. (2020) opted for an increasing, linear schedule $\beta_1 = 10^{-4}$ to $\beta_T =$

0.02, which was later replaced by a cosine schedule in an improved version presented in a follow-up work (Dhariwal and Nichol, 2021b). A fixed noise schedule yields the important property that every latent $\boldsymbol{x}_t$ for $t \in [1, T]$ can be computed in closed form. Given a latent $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})$, it can also be expressed as:

$$\begin{aligned}
\boldsymbol{x}_t &= \sqrt{1-\beta_t}\boldsymbol{x}_{t-1} + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{1-\beta_t}\left(\sqrt{1-\beta_{t-1}}\boldsymbol{x}_{t-2} + \sqrt{\beta_{t-1}}\boldsymbol{\epsilon}_{t-2}\right) + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{(1-\beta_t)(1-\beta_{t-1})}\boldsymbol{x}_{t-2} + \sqrt{(1-\beta_t)\beta_{t-1}}\boldsymbol{\epsilon}_{t-2} + \sqrt{\beta_t}\boldsymbol{\epsilon}_{t-1} \\
&= \sqrt{(1-\beta_t)(1-\beta_{t-1})}\boldsymbol{x}_{t-2} + \sqrt{1-(1-\beta_t)(1-\beta_{t-1})}\boldsymbol{\epsilon}
\end{aligned} \tag{21}$$

Note that in the last line of Equation (21), two Gaussians $\mathcal{N}(\boldsymbol{\epsilon}_{t-1}; \mathbf{0}, \beta_t \boldsymbol{I})$ and $\mathcal{N}(\boldsymbol{\epsilon}_{t-1}; \mathbf{0}, (1-\beta_t)\beta_{t-1}\boldsymbol{I})$ are merged. This is accomplished by adding the variances:

$$\begin{aligned}
&\beta_t + (1-\beta_t)\beta_{t-1} = \beta_t + \beta_{t-1} - \beta_t\beta_{t-1} \\
&= 1 - (1 - \beta_t - \beta_{t-1} + \beta_t\beta_{t-1}) = 1 - (1-\beta_t)(1-\beta_{t-1})
\end{aligned}$$

which yields the merged Gaussian $\mathcal{N}(\boldsymbol{\epsilon}; \mathbf{0}, 1-(1-\beta_t)(1-\beta_{t-1})\boldsymbol{I})$. Recursively applying such merged Gaussians as in Equation (21) results in the following closed form posterior:

$$\begin{aligned}
q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) &= \mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_0, (1-\alpha_t)\boldsymbol{I}\right) \text{ with} \\
\alpha_t &\coloneqq \prod_{s=1}^{t}(1-\beta_s).
\end{aligned} \tag{22}$$

Therefore, every latent can be expressed in terms of the data sample $\boldsymbol{x}_0 \in \mathcal{D}$ and random, sampled noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ as:

$$\boldsymbol{x}_t = \sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon} \;\; \forall t \in [1, T]. \tag{23}$$

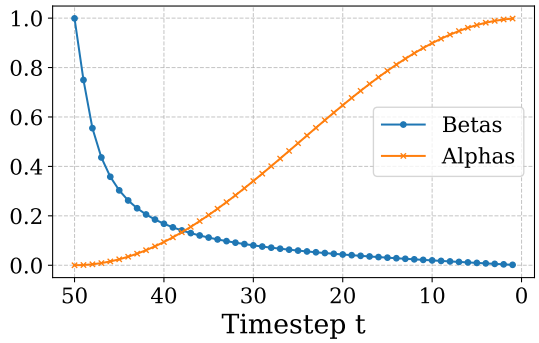
By first inspection of Equation (23), it resembles the fundamental idea of Equation (1) with $\boldsymbol{x}_0$ begin the signal and 'artificially' added, sampled noise $\boldsymbol{\epsilon}$. In this context, many authors also consider the (log-)SNR:

$$\mathrm{SNR}(t) \coloneqq \log\left(\frac{\alpha_t}{1-\alpha_t}\right) \tag{24}$$

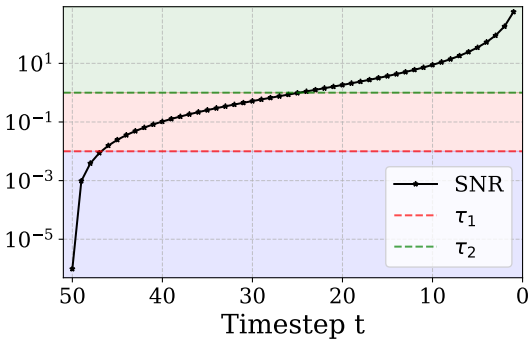
graphically depiced in Figure 2b for a corresponding cosine schedule of the variances $\beta_{[1:T]}$ as defined in Dhariwal and Nichol (2021b). These in turn are illustrated together with the resulting schedule for $\alpha_{[1:T]}$ in Figure 2a. As is evident, the cosine schedule causes the noise to diminish faster compared to a linear schedule. This was hypothesized to provide the model with easier learning tasks earlier on in the process (Dhariwal and Nichol, 2021b), which we will elaborate more in Section 4.2.

The actual training objective for a DPPM is obtained by simultaneously minimizing the ELBO between corresponding forward and reverse kernels, similar to the approach for VAEs in Equation (15) for a single pair of such kernels. Ho et al. (2020) further observed that stability during training can be significantly improved by explicitly conditioning the forward kernels on their origin, i.e., the datapoint $\boldsymbol{x}_0$. By Bayes' rule:

$$q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) = q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) = \frac{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)} \tag{25}$$



(a) Cosine variance schedule $\beta_{[1:T]}$ and corresponding signal schedule $\alpha_{[1:T]}$.



(b) (log-)SNR corresponding to the variance schedule from Figure 2a.

Figure 2: Illustration of the resulting signal and noise components for a DDPM as defined in (Dhariwal and Nichol, 2021b). For a more meaningful interpretation, recall that latents exhibit values in the range of [-1,1] by design. Color codings as well as τ_1 and τ_2 in Figure 2b are relevant for Section 4.2 and mark different stages of the generative process.

where the first equality holds due to the Markov property. This yields the following reformulation of the ratio between the joint and posterior distributions, which will subsequently be used to derive the DDPM training objective:

$$
\begin{aligned}
&\log\left(\frac{p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{[0:T]}\right)}{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}\right) \\
&\overset{(19),(20)}{=} \left(\log p(\boldsymbol{x}_T) + \sum_{t=1}^{T} \log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})}\right)\right) \\
&= \log p(\boldsymbol{x}_T) + \log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}\right) + \sum_{t=2}^{T} \log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1})}\right) \\
&\overset{(25)}{=} \log\left(\frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)}\right) + \sum_{t=2}^{T} \log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)}\right)
\end{aligned}
\tag{26}
$$

As evident from the last sum in Equation (26) and schematically illustrated in Figure 1, the reverse kernels are compared against the inverse forward kernels, additionally conditioned on $\boldsymbol{x}_0$. Now, the ELBO of the DDPM can be derived as follows by marginalizing over latents using the forward posterior $q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)$ and lower bounding the evidence similar

to Equation (14):

$$
\begin{aligned}
\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0) &= \log \int_{\mathcal{X}} p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{[0:T]}\right) d\boldsymbol{x}_{[1:T]} = \log \int_{\mathcal{X}} p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{[0:T]}\right) \frac{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)} d\boldsymbol{x}_{[1:T]} \\
&= \log \mathbb{E}_{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}\left[\frac{p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{[0:T]}\right)}{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}\right] \geq \mathbb{E}_{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}\left[\log\left(\frac{p_{\boldsymbol{\theta}}\left(\boldsymbol{x}_{[0:T]}\right)}{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}\right)\right] \\
&\overset{(26)}{=} \mathbb{E}_{q\left(\boldsymbol{x}_{[1:T]} \mid \boldsymbol{x}_0\right)}\left[\log\left(\frac{p(\boldsymbol{x}_T) p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)}\right) + \sum_{t=2}^{T} \log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)}\right)\right] \\
&= \mathbb{E}_{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)\right] + \mathbb{E}_{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)}\left[\log\left(\frac{p(\boldsymbol{x}_T)}{q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)}\right)\right] \\
&\quad + \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t, \boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}\left[\log\left(\frac{p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)}{q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)}\right)\right] \\
&= \underbrace{\mathbb{E}_{q(\boldsymbol{x}_1 \mid \boldsymbol{x}_0)}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)\right]}_{\text{reconstruction term}} - \underbrace{\mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{x}_T \mid \boldsymbol{x}_0) \parallel p(\boldsymbol{x}_T))}_{\text{prior matching term}} \\
&\quad \underbrace{- \sum_{t=2}^{T} \mathbb{E}_{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)}\left[\mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]}_{\text{denoising matching term}}
\end{aligned}
\tag{27}
$$

Note that, if $T = 1$, this formulation essentially resembles the vanilla VAE ELBO from Equation (14). The goal is to maximize the lower bound, which is achieved by simultaneously maximizing the reconstruction term and minimizing the prior and denoising matching terms. Furthermore, the objective was derived only leveraging the Markov property, making it generally applicable to MHVAEs. As before, the reconstruction term can be approximated using Monte-Carlo estimates. In the DDPM framework, the final decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)$ is a custom implementation suitable for modeling discrete image data, which will be elaborated further at the end of this section. The prior matching term is not optimizable and (very close to) zero since both the prior $p(\boldsymbol{x}_T)$ and posterior $q(\boldsymbol{x}_T \mid \boldsymbol{x}_0)$ were chosen to have very similar functional forms (i.e., standard Gaussians). The dominant factor in this ELBO derivation is the denoising matching term, whose estimation requires repeated sampling of latents $\boldsymbol{x}_t \sim q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)$. However, these can be obtained very efficiently for any $t \in [1, T]$ and $\boldsymbol{x}_0 \in \mathcal{D}$ in closed form using Equation (23).

We will now focus exclusively on this denoising matching term to derive the loss function that serves as a differentiable signal for training the DDPM. As is evident, the goal is to model the transitioning kernels $q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0)$. To obtain a closed representation of these, we re-formulate Equation (25) as follows:

$$
\begin{aligned}
q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) &= \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}, \boldsymbol{x}_0) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)} = \frac{q(\boldsymbol{x}_t \mid \boldsymbol{x}_{t-1}) q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_0)}{q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)} \\
&\overset{(19),(22)}{=} \frac{\mathcal{N}\left(\boldsymbol{x}_t; \sqrt{1-\beta_t}\boldsymbol{x}_0, \beta_t \boldsymbol{I}\right) \mathcal{N}\left(\boldsymbol{x}_{t-1}; \sqrt{\alpha_{t-1}}\boldsymbol{x}_0, (1-\alpha_{t-1})\boldsymbol{I}\right)}{\mathcal{N}\left(\boldsymbol{x}_t; \sqrt{\alpha_t}\boldsymbol{x}_0, (1-\alpha_t)\boldsymbol{I}\right)} \\
&\propto \mathcal{N}\Bigg(\boldsymbol{x}_{t-1}; \underbrace{\frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \sqrt{\alpha_{t-1}}\beta_t \boldsymbol{x}_0}{1-\alpha_t}}_{\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)}, \underbrace{\frac{(1-\alpha_{t-1})}{1-\alpha_t}\beta_t \boldsymbol{I}}_{\boldsymbol{\Sigma}_q(t)}\Bigg)
\end{aligned}
\tag{28}
$$

For a detailed derivation of the last Gaussian, we refer to Luo (2022). Note that occasionally we also abbreviate as:

$$\boldsymbol{\Sigma}_q(t) = \sigma_q^2(t)\boldsymbol{I} \text{ with } \sigma_q^2(t) \coloneqq \frac{(1-\alpha_{t-1})}{1-\alpha_t}\beta_t. \tag{29}$$

Ho et al. (2020) note that the simplified choice of $\sigma_q^2(t) = \beta_t$ yielded similar results (see also Section 2.4.6). As the KL-divergence measures the similarity between two distributions, it is intuitive to also choose a Gaussian as the functional form for the learned reverse kernels:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t) \coloneqq \mathcal{N}\left(\boldsymbol{x}_t; \mu_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t), \Sigma_{\boldsymbol{\theta}}(t)\right) \tag{30}$$

Since the variance of the forward process in Equation (29) is only conditioned on the predefined noise schedule $\beta_{[1:T]}$, it can be computed very efficiently even during the reverse process. Consequently, Ho et al. (2020) chose to set the same fixed variance $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(t) \coloneqq \boldsymbol{\Sigma}_q(t)$ for the reverse process. Note that learned variance schedules were later shown to perform similarly well with certain modifications (Dhariwal and Nichol, 2021a; Dhariwal and Nichol, 2021b; Diederik Kingma et al., 2021). Furthermore, the predicted posterior mean $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ should be as similar as possible to the 'true' posterior mean $\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$. However, the former cannot condition on real data $\boldsymbol{x}_0$ since the reverse process is supposed to generate new data from completely random noise. Therefore, it was defined as:

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \coloneqq \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \sqrt{\alpha_{t-1}}\beta_t\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1-\alpha_t}. \tag{31}$$

This closely matches the mean $\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$ in Equation (28). Here, $\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ is the output of a deep neural network that receives the current latent $\boldsymbol{x}_t$ and timestep t as input. The reverse transitioning kernel (30) can thus be interpreted as $p(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0 \approx \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t))$. This formulation can be leveraged to yield a suitable regression target for the neural network by analyzing the denoising matching term from Equation (27). To this end, note that the KL-divergence between two Gaussians is given in a closed analytical form as:

$$\begin{aligned} &\mathrm{D}_{\mathrm{KL}}\left(\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_{\boldsymbol{x}}, \boldsymbol{\Sigma}_{\boldsymbol{x}}) \parallel \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{y}}, \boldsymbol{\Sigma}_{\boldsymbol{y}})\right) \\ &= \frac{1}{2}\left[\log\frac{|\boldsymbol{\Sigma}_{\boldsymbol{y}}|}{|\boldsymbol{\Sigma}_{\boldsymbol{x}}|} - d + \operatorname{tr}\left(\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}\boldsymbol{\Sigma}_{\boldsymbol{x}}\right) + (\boldsymbol{\mu}_{\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{x}})^T\boldsymbol{\Sigma}_{\boldsymbol{y}}^{-1}(\boldsymbol{\mu}_{\boldsymbol{y}} - \boldsymbol{\mu}_{\boldsymbol{x}})\right] \end{aligned} \tag{32}$$

where d is the dimensionality of the dataspace $\mathcal{X}$ (Luo, 2022). Defining $\boldsymbol{\mu}_{\boldsymbol{\theta}} \coloneqq \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$

and $\boldsymbol{\mu}_q \coloneqq \boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$ for notational clarity, we can derive the objective as:

$$
\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} \mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2}\left[\log\left(\frac{|\boldsymbol{\Sigma}_q|}{|\boldsymbol{\Sigma}_q|}\right) - d + \operatorname{tr}\left(\boldsymbol{\Sigma}_q^{-1}\boldsymbol{\Sigma}_q\right) + (\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q)\left(\boldsymbol{\Sigma}_q\right)^{-1}(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q)\right] \\
&\overset{(29),(32)}{=} \arg\min_{\boldsymbol{\theta}} \frac{1}{2}\left[\log(1) - d + d + (\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q)\left(\sigma_q^2(t)\boldsymbol{I}\right)^{-1}(\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q)\right] \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \left\|\boldsymbol{\mu}_{\boldsymbol{\theta}} - \boldsymbol{\mu}_q\right\|_2^2 \qquad (33) \\
&\overset{(28),(31)}{=} \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \left\| \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \sqrt{\alpha_{t-1}}\beta_t \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{1-\alpha_t} \right. \\
&\qquad \left. - \frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \sqrt{\alpha_{t-1}}\beta_t \boldsymbol{x}_0}{1-\alpha_t} \right\|_2^2 \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \left\| \frac{\sqrt{\alpha_{t-1}}(\beta_t)}{1-\alpha_t}(\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0) \right\|_2^2 \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{\alpha_{t-1}\beta_t^2}{(1-\alpha_t)^2} \left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right\|_2^2 \\
&\overset{(29)}{=} \arg\min_{\boldsymbol{\theta}} \frac{\alpha_{t-1}\beta_t}{2(1-\alpha_{t-1})(1-\alpha_t)} \left\|\hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{x}_0\right\|_2^2 \qquad (34)
\end{aligned}
$$

Examining Equation (34), it becomes evident that optimizing a DDPM involves training a neural network to recover the original ground truth image from an arbitrarily noisified version $\boldsymbol{x}_t$ of it. An alternative, asymptotically equivalent objective in Equation (33) is to train a neural network $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ to predict the forward process' posterior mean $\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)$ where both objectives are conntected through Equation (31). Given a discrete timestep t, these objectives differ only by a constant factor determined by the predefined variances $\beta_{[1:T]}$. Despite the viability of these objectives for training, Ho et al. (2020) derived yet another regression target for the neural network, which yielded superior results in practice. To this end, Equation (23) is reformulated as:

$$
\boldsymbol{x}_0 = \frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}}{\sqrt{\alpha_t}}. \qquad (35)
$$

Furthermore, note that by Equation (22), we have:

$$
\frac{\alpha_{t-1}}{\alpha_t} = \frac{\prod_{s=1}^{t-1}(1-\beta_s)}{\prod_{s=1}^{t}(1-\beta_s)} = \frac{1}{1-\beta_t}. \qquad (36)
$$

Thus, the true transitioning mean $\boldsymbol{\mu}_q$ from Equation (28) can be expressed without con-

ditioning on $\boldsymbol{x}_0$ as follows:

$$\begin{aligned}
\boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0) &\overset{(35)}{=} \boldsymbol{\mu}_q\left(\boldsymbol{x}_t, \frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}}{\sqrt{\alpha_t}}\right) \\
&\overset{(28)}{=} \frac{1}{1-\alpha_t}\left(\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \frac{\sqrt{1-\alpha_t}\beta_t\left(\boldsymbol{x}_t - \sqrt{1-\alpha_t}\boldsymbol{\epsilon}\right)}{\sqrt{\alpha_t}}\right) \\
&= \frac{1}{1-\alpha_t}\left(\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \sqrt{\frac{\alpha_{t-1}}{\alpha_t}}\beta_t\boldsymbol{x}_t - \sqrt{\frac{\alpha_{t-1}(1-\alpha_t)}{\alpha_t}}\beta_t\boldsymbol{\epsilon}\right) \\
&\overset{(36)}{=} \frac{1}{1-\alpha_t}\left(\sqrt{1-\beta_t}(1-\alpha_{t-1})\boldsymbol{x}_t + \frac{\beta_t}{\sqrt{1-\beta_t}}\boldsymbol{x}_t - \sqrt{\frac{1-\alpha_t}{1-\beta_t}}\beta_t\boldsymbol{\epsilon}\right) \\
&= \left(\frac{\sqrt{1-\beta_t}(1-\alpha_{t-1})}{1-\alpha_t} + \frac{\beta_t}{\sqrt{1-\beta_t}(1-\alpha_t)}\right)\boldsymbol{x}_t - \frac{\sqrt{1-\alpha_t}\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\boldsymbol{\epsilon} \\
&= \left(\frac{(1-\beta_t)(1-\alpha_{t-1}) + \beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\right)\boldsymbol{x}_t - \frac{\sqrt{1-\alpha_t}\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\boldsymbol{\epsilon} \\
&= \left(\frac{1-\beta_t+\alpha_{t-1}(1-\beta_t)+\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\right)\boldsymbol{x}_t - \frac{\sqrt{1-\alpha_t}\beta_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\boldsymbol{\epsilon} \\
&= \frac{1-\alpha_t}{(1-\alpha_t)\sqrt{1-\beta_t}}\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\boldsymbol{\epsilon} \\
&= \frac{1}{\sqrt{1-\beta_t}}\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\boldsymbol{\epsilon}
\end{aligned} \tag{37}$$

Once again, we set the learned reverse process mean accordingly:

$$\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \frac{1}{\sqrt{1-\beta_t}}\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \tag{38}$$

where $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \approx \boldsymbol{\epsilon}$ defines a deep neural network used to predict the noise in the generative process that was used to obtain latent $\boldsymbol{x}_t$ from $\boldsymbol{x}_0$. The new objective can therefore be derived by substituting both means in Equation (33) as follows:

$$\begin{aligned}
&\arg\min_{\boldsymbol{\theta}} \mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t)) \\
&\overset{(33)}{=} \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \left\|\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \boldsymbol{\mu}_q(\boldsymbol{x}_t, \boldsymbol{x}_0)\right\|_2^2 \\
&\overset{(37),(38)}{=} \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \left\|\frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\boldsymbol{\epsilon} - \frac{\beta_t}{\sqrt{1-\alpha_t}\sqrt{1-\beta_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\right\|_2^2 \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2\sigma_q^2(t)} \frac{\beta_t^2}{(1-\alpha_t)(1-\beta_t)} \left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\right\|_2^2 \\
&= \arg\min_{\boldsymbol{\theta}} \frac{\beta_t}{2(1-\alpha_{t-1})(1-\beta_t)} \left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\right\|_2^2
\end{aligned} \tag{39}$$

Consequently, a DDPM can be trained by learning a neural network to predict either the transition kernels' means in the forward process (33), the original images (34) or the noise employed in the diffusion process (39). Although all three regression targets are equally valid and in theory differ only by time-dependent constants, Ho et al. (2020)

empirically found the noise prediction Objective (39) to yield samples of superior quality in practice compared to both other choices. Furthermore, it was found beneficial for sample quality to omit the time-dependent constants in the final formulation of Objective (39). This was interpreted as a re-weighting of the ELBO which causes greater losses for timesteps closer to T (Ho et al., 2020). Therefore, emphasis was placed on tasks which are presumably harder to learn by means of human perception due to the advanced deterioration of the samples (see also Section 4.2). Generally, referring back to the ELBO in Equation (27), Objective (39) must be optimized for $t \in [2, T]$. Ho et al. (2020) advised to do so by computing the expectation over uniformly sampled timesteps $t \sim \mathcal{U}\{2, T\}$ using a single set of parameters $\boldsymbol{\theta}$ to jointly model all reverse kernels using a deep neural network conditioned on the current timestep. Together with the closed form derivation of the latents from Equation (23) and the simplified Objective (39), the final tractable loss function or learning signal with respect to the parameters is defined as:

$$\begin{aligned}
\mathcal{L}_{\text{simple}}(\boldsymbol{\theta}) &\overset{(39)}{\coloneqq} \mathbb{E}_{t\sim\mathcal{U}\{2,T\}}\left[\mathbb{E}_{\boldsymbol{x}_t\sim q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\mathrm{D}_{\mathrm{KL}}(q(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t, \boldsymbol{x}_0) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{x}_{t-1} \mid \boldsymbol{x}_t))\right]\right] \\
&\overset{(23)}{=} \mathbb{E}_{t\sim\mathcal{U}\{2,T\}}\left[\mathbb{E}_{\boldsymbol{x}_0\in\mathcal{D},\boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}\left(\sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, t\right)\right\|_2^2\right]\right] \\
&= \mathbb{E}_{t,\boldsymbol{x}_0,\boldsymbol{\epsilon}}\left[\left\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}\left(\sqrt{\alpha_t}\boldsymbol{x}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, t\right)\right\|_2^2\right]
\end{aligned} \tag{40}$$

This term is easily tractable and can be efficiently approximated using stochastic samples of the data and Monte-Carlo estimates over uniformly sampled noise levels (or timesteps). For $t = 1$, the final decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)$ is trained through the loss signal of the reconstruction term in Equation (27). Ho et al. (2020) chose to model latents $\boldsymbol{x}_{[1:T]}$ using values in the range of $[-1, 1]$. Therefore, the authors proposed a custom defined decoder $p_{\boldsymbol{\theta}}(\boldsymbol{x}_0 \mid \boldsymbol{x}_1)$, which is theoretically capable of a lossless and noiseless output for discrete image data with values in $[0, 255]^3$ when $T \to \infty$. For a detailed description of this custom decoder, readers are referred to Ho et al. (2020).

2.4.2 Score-Based Generative Models

While MLE (see Equation (4)) represents one specific approach to generative modeling of the 'true' data PDF $p(\boldsymbol{x})$, alternative formulations exists. Assuming this PDF can indeed be modeled, every parametric attempt to define an approximation $p_{\boldsymbol{\theta}}(\boldsymbol{x}) \approx p(\boldsymbol{x})$ must adhere to the first axiom of PDFs, which states that $\int_{\mathcal{X}} p_{\boldsymbol{\theta}}(\boldsymbol{x}) d\boldsymbol{x} = 1$. This is conceptually equivalent to seeking an unnormalized likelihood model $\tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x})$ with:

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{\tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x})}{\int_{\mathcal{X}} \tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) d\boldsymbol{x}} \tag{41}$$

As previously noted in Section 2, such an integral in Equation (41), denoted as the *normalization constant*, often presents a significant computational challenge in practice. A long-standing approach to circumvent this issue involves applying the gradient of the log-

density in data space, formally expressed as:

$$\begin{aligned} s_{\boldsymbol{\theta}}(\boldsymbol{x}) &\coloneqq \nabla_{\boldsymbol{x}} \log p_{\boldsymbol{\theta}}(\boldsymbol{x}) \overset{(41)}{=} \nabla_{\boldsymbol{x}} \log \left(\frac{\tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x})}{\int_{\mathcal{X}} \tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) d\boldsymbol{x}} \right) \\ &= \nabla_{\boldsymbol{x}} \log \tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) - \underbrace{\nabla_{\boldsymbol{x}} \log \int_{\mathcal{X}} \tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) d\boldsymbol{x}}_{=0} \\ &= \nabla_{\boldsymbol{x}} \log \tilde{p}_{\boldsymbol{\theta}}(\boldsymbol{x}) \end{aligned} \tag{42}$$

The second term in the second line of Equation (42) vanishes because, after integrating over the whole data space $\mathcal{X}$, the resulting value is a constant with respect to $\boldsymbol{x}$. This property was first discovered by Fisher (1935), who coined $s(\boldsymbol{x}) \coloneqq \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$ as the *score function*. It can be interpreted as a vector field in data space pointing to regions of higher data log-likelihoods.

Due to the interesting property of Equation (42), the score function has been of special interest to the generative modeling community. A first, somewhat tractable approach was provided by Hyvärinen (2005). Similar to the simple denoising loss (40), a suitable loss function to learn a model $s_{\boldsymbol{\theta}}(\boldsymbol{x})$ of the true score $s(\boldsymbol{x})$ would be:

$$\mathcal{L}_{SM}(\boldsymbol{\theta}) \coloneqq \frac{1}{2} \mathbb{E}_{p(\boldsymbol{x})} \left[\| s_{\boldsymbol{\theta}}(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) \|_2^2 \right]. \tag{43}$$

Optimizing using Equation (43) is known as (simple) *score matching* (Hyvärinen, 2005). However, $\mathcal{L}_{SM}(\boldsymbol{\theta})$ is not tractable due to the inaccessibility of $\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x})$. The main contribution of Hyvärinen (2005) was to uncover the equivalence (under mild regularity conditions on $s_{\boldsymbol{\theta}}$) of Equation (43) to:

$$\mathcal{L}_{ISM}(\boldsymbol{\theta}) \coloneqq \mathbb{E}_{p(\boldsymbol{x})} \left[\frac{1}{2} \| s_{\boldsymbol{\theta}}(\boldsymbol{x}) \|_2^2 + Tr\left(\nabla_{\boldsymbol{x}} s_{\boldsymbol{\theta}}(\boldsymbol{x}) \right) \right]. \tag{44}$$

The remarkable property of optimization with Equation (44), which is coined *implicit score matching*, is the absence of the intractable true score $s(\boldsymbol{x})$. However, two major problems remain with this approach. Firstly, computing the trace of the Jacobian, i.e., $Tr(\nabla_{\boldsymbol{x}} s_{\boldsymbol{\theta}}(\boldsymbol{x}))$, can quickly become a serious bottleneck, especially when $s_{\boldsymbol{\theta}}(\boldsymbol{x})$ is parametrized with deep neural networks. Secondly, the objective of modeling scores over the whole data space $\mathcal{X}$ to enable generation of data from random samples is not possible when optimizing only under the expectation of real data samples $\mathbb{E}_{p(\boldsymbol{x})}$. Vincent (2011) first discovered a principled approach with theoretical guarantees to solve these problems. Therefore, Equation (43) was modified to:

$$\mathcal{L}_{DSM}(\boldsymbol{\theta}) \coloneqq \mathbb{E}_{p^{\sigma}(\tilde{\boldsymbol{x}}, \boldsymbol{x})} \left[\| s_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}) - \nabla_{\tilde{\boldsymbol{x}}} \log p^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) \|_2^2 \right] \tag{45}$$

with $p^{\sigma}(\tilde{\boldsymbol{x}}, \boldsymbol{x}) \coloneqq p^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) p(\boldsymbol{x})$ where $\tilde{\boldsymbol{x}} \sim p^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x})$ is a noised version of the data sample $\boldsymbol{x} \sim p(\boldsymbol{x})$ according to some noise model (e.g., Gaussian) with standard deviation σ. This approach solves both aforementioned problems as $\tilde{\boldsymbol{x}} \sim p^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) p(\boldsymbol{x})$ yields samples over a far greater domain by noisification and $p^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x})$ can usually be evaluated directly, similar to the Gaussian forward kernels in a DDPM. Vincent (2011) discovered that a Gaussian noise kernel $p_{\mathcal{N}}^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) \coloneqq \mathcal{N}(\boldsymbol{x}, \sigma^2 \boldsymbol{I})$ yields:

$$\nabla_{\tilde{\boldsymbol{x}}} \log p_{\mathcal{N}}^{\sigma}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) = -\frac{1}{\sigma^2} (\tilde{\boldsymbol{x}} - \boldsymbol{x}). \tag{46}$$

This can be interpreted as a unit vector from the corrupted sample $\tilde{\boldsymbol{x}}$ toward the real sample $\boldsymbol{x}$. As such, Equation (46), which is what is being learned by the score model in $\mathcal{L}_{DSM}(\boldsymbol{\theta})$, denotes the noise term used to corrupt the data sample. This was considered the first connection between score-based and denoising frameworks. To make the connection between score prediction and noise prediction objectives from NCSN and DDPM explicit, recall the closed form forward kernel from Section 2.4.1 as $q(\boldsymbol{x}_t \mid \boldsymbol{x}_0) \sim \mathcal{N}(\sqrt{\alpha_t}\boldsymbol{x}_0, (1 - \alpha_t)\boldsymbol{I})$. In this framework, the score is constructed over the marginals of each latent as given by the forward process, i.e., $s_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \approx \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t)$. Therefore, it is:

$$\begin{aligned} s_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \approx \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t) &= \mathbb{E}_{p(\boldsymbol{x}_0)} \left[\nabla_{\boldsymbol{x}_t} q(\boldsymbol{x}_t \mid \boldsymbol{x}_0)\right] \\ &= \mathbb{E}_{p(\boldsymbol{x}_0)} \left[-\frac{\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{\sqrt{1-\alpha_t}}\right] = -\frac{\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{\sqrt{1-\alpha_t}} \end{aligned} \tag{47}$$

Furthermore, $p^{\sigma}_{\mathcal{N}}$ is typically easily tractable, which allows for efficient Monte-Carlo estimations over data samples to enable optimization using Equation (45). Moreover, Vincent (2011) proved that $\mathcal{L}_{DSM}(\boldsymbol{\theta})$ is guaranteed to lead to an unbiased estimate of the true score. However, in this regard, σ yields a certain trade-off that must be accounted for. If it is too small, the scores may quickly become inaccurate for domains outside of $\boldsymbol{x} \sim p(\boldsymbol{x})$, and if it is too large, it may enforce learning a degraded, noised version of the true data distribution $p(\boldsymbol{x})$.

It wasn't until Y. Song and Ermon (2019) that this approach reached state-of-the-art generative results by accounting for the aforementioned trade-off in a suitable, stable manner. The authors chose to model T different noise levels $\sigma_1 > \ldots > \sigma_T$ such that $p^{\sigma_T}(\tilde{\boldsymbol{x}}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and $p^{\sigma_1}(\tilde{\boldsymbol{x}}) = p(\boldsymbol{x})$. The score function is learned jointly over all noise levels with the respective conditioning on σ, i.e., $s_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}, \sigma)$. The loss objective was given as:

$$l(\boldsymbol{\theta}, \sigma) \coloneqq \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}\mathbb{E}_{\tilde{\boldsymbol{x}}\sim p^{\sigma}(\tilde{\boldsymbol{x}}|\boldsymbol{x})}\left[\left\| s_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}, \sigma) + \frac{\tilde{\boldsymbol{x}} - \boldsymbol{x}}{\sigma^2}\right\|_2^2\right]$$

per noise level with the following unified objective:

$$\mathcal{L}_{NCSN}\left(\boldsymbol{\theta}, \sigma_{[1:T]}\right) \coloneqq \frac{1}{T}\sum_{t=1}^{T} \lambda(\sigma_t) l(\boldsymbol{\theta}, \sigma_t) \tag{48}$$

where $\lambda(\sigma_t)$ is a coefficient function. Using deep neural networks to model $s_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma_t)$ together with Equation (48) as a learning signal, is termed a *noise conditional score network (NCSN)* and, upon closer inspection, bears many resemblances to DDPMs. This connection was made explicit in a follow-up work by Y. Song et al. (2021) under the framework of *stochastic differential equations (SDEs)* by comparing both frameworks in the limit of $T \to \infty$ with infinitesimal change between noise levels. Due to the limited scope of this work and the relevance of SDEs therein, this perspective is omitted here. However, it should be noted that this line of research has led to the conclusion that all contemporary diffusion-type objectives, including NCSNs and DDPMs, are equivalent to a weighted integral of ELBOs over different noise levels (Diederik P Kingma and Gao, 2023). In that sense, adding Gaussian noise is simply one special type of data augmentation that can be used to obtain an estimate on the ELBO. In fact, considering f.e. a completely black data point and a Gaussian diffusion would deteriorate the sample in the quickest way possible

compared to other parametrized PDFs of same variance. This is due to the fact that it encodes the least prior information or structure, hence why it is called entropy maximizing (Heaton, 2017, §19.4.1). It has since been shown that other types of diffusion can perform better, depending on the domain of the data to be modeled (Nachmani et al., 2021).

Lastly, it is important to elucidate how these frameworks, DDPM and NCSN, were typically sampled before the advent of the SDE formulation. Recall that the score $s_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma)$ is supposed to point in the direction of higher data (log-)likelihoods at each noise level. Consequently, a greedy approach to yielding subsequent latents is by recursively applying:

$$\boldsymbol{x}_{t-1} = \boldsymbol{x}_t + \zeta_t s_{\boldsymbol{\theta}}(\boldsymbol{x}_t, \sigma_t) + \sqrt{2\zeta_t}\boldsymbol{\epsilon} \text{ with } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}) \tag{49}$$

for $t \in [1, T]$ in a Markov chain of updates using only the learned score $s_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma)$ and starting from random noise $\boldsymbol{x}_T \sim p^{\sigma_T}(\tilde{\boldsymbol{x}}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The coefficients $\zeta_{[1:T]}$ were chosen to be a gradually decreasing step size of $\zeta_t \propto \sigma_t^2$. For a more detailed explanation of specifically Equation (49), readers are referred to Y. Song and Ermon (2019). This approach was termed *stochastic gradient Langevin dynamics (SGLD)*, a combination of SGD techniques (see Section 2.1) and the physical concept of Langevin dynamics, proposed by Welling and Teh (2011). The added noise stems from the latter and is intended to prevent the algorithm from becoming stuck in local minima of the learned loss function's surface. It was similarly adapted in the subsequent work of Ho et al. (2020) for DDPMs where the exact transitioning equation amounted to:

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}}\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}}\right) + \sqrt{\beta_t}\boldsymbol{\epsilon} \text{ with } \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}). \tag{50}$$

Essentially, this is equivalent to sampling from the closed posterior of the forward process given by Equation (22) using the predicted $\boldsymbol{x}_0$ given by Equation (35) and additional, scaled noise. For a more detailed explanation of the parameters of Equation (50), readers are referred to Ho et al. (2020).

2.4.3 Guided Sampling

Despite their impressive results (Y. Song and Ermon, 2019; Ho et al., 2020; Dhariwal and Nichol, 2021b; Y. Song et al., 2021), diffusion models lacked the flexibility to effectively condition the generative process on certain classes present in the training dataset, formally denoted as:

$$\mathcal{D} \coloneqq \{(\boldsymbol{x}^n, c^n)\}_{n=1}^N.$$

The first rather successful approach, termed *classifier guidance*, was proposed by Dhariwal and Nichol (2021a). The authors trained a classifier $f_{\boldsymbol{\Phi}}(c \mid \boldsymbol{x}_t, t)$ on noisy latents $\boldsymbol{x}_t$ over all timesteps and use the score of these $\nabla_{\boldsymbol{x}} \log f_{\boldsymbol{\Phi}}(c \mid \boldsymbol{x}_t, t)$ to guide towards samples during generation that are more faithful to class c. This is consistent with the intuition about the score as described in the previous section. From this perspective, the score of the joint of the forward process $\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t, c)$ is of interest, which can be approximated using Baye's

rule $q(\boldsymbol{x}_t, c) = q(\boldsymbol{x}_t)q(\boldsymbol{x}_t \mid c)$ as:

$$\begin{aligned}\nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t, c) &= \nabla_{\boldsymbol{x}_t} \log q(\boldsymbol{x}_t) + \nabla_{\boldsymbol{x}_t} \log q(c \mid \boldsymbol{x}_t) \\ &\overset{(47)}{\approx} -\frac{1}{\sqrt{1-\alpha_t}}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) + \nabla_{\boldsymbol{x}_t} \log f_{\boldsymbol{\Phi}}(c \mid \boldsymbol{x}_t) \\ &= -\frac{1}{\sqrt{1-\alpha_t}}\left(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \sqrt{1-\alpha_t}\nabla_{\boldsymbol{x}_t} \log f_{\boldsymbol{\Phi}}(c \mid \boldsymbol{x}_t)\right).\end{aligned} \tag{51}$$

The last line of Equation (51) implies that updating a noise prediction greedily in a recursive chain given as:

$$\bar{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) \coloneqq \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) - \omega\sqrt{1-\alpha_t}\nabla_{\boldsymbol{x}_t} \log f_{\boldsymbol{\Phi}}(c \mid \boldsymbol{x}_t) \tag{52}$$

using some suitable step size $\omega \in \mathbb{R}$ should yield a sample $\boldsymbol{x}_0$ faithful to class c. However, this approach requires training of custom classifiers over all noise levels, which can quickly become a serious complexity requirement (Ho and Salimans, 2021).

In a follow-up work, Ho and Salimans (2021) proposed a very simple yet effective approach to achieve high-quality conditional data without training any external classifiers. They proposed to jointly train two models, an unconditional $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ and a conditional $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c)$ model, using the same deep neural network (parameters $\boldsymbol{\theta}$). The former is obtained by simply dropping the conditioning of samples periodically, such that the model learns to generate samples unconditionally, i.e., $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t) = \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c = \emptyset)$. Now, by Bayes' rule:

$$p(c \mid \boldsymbol{x}_t) = \frac{p(\boldsymbol{x}_t \mid c)p(c)}{p(\boldsymbol{x}_t)} \tag{53}$$

and therefore, by applying the score:

$$\begin{aligned}\nabla_{\boldsymbol{x}_t} \log p(c \mid \boldsymbol{x}_t) &\overset{(53)}{=} \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t \mid c) - \nabla_{\boldsymbol{x}_t} \log p(\boldsymbol{x}_t) + \underbrace{\nabla_{\boldsymbol{x}_t} \log p(c)}_{=0} \\ &\overset{(47)}{\approx} -\frac{1}{\sqrt{1-\alpha_t}}\left(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\right)\end{aligned} \tag{54}$$

Using Equation (51) and the reformulated term from Equation (54) yields:

$$\begin{aligned}\bar{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) &= \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \omega\sqrt{1-\alpha_t}\nabla_{\boldsymbol{x}_t} \log p(c \mid \boldsymbol{x}_t) \\ &\overset{(54)}{\approx} \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) + \omega\left(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\right) \\ &= (\omega+1)\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t, c) - \omega\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)\end{aligned} \tag{55}$$

Note, however, that this derivation is merely a theoretical inspiration since $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ is modeled as a deep neural network and therefore not guaranteed to be the gradient of any classifier, if at all. We further discuss this point in Section 5.1.2. Empirically, however, the approach in Equation (55) has yielded impressive generative results (Ho and Salimans, 2021) and has since become the de facto standard for conditional diffusion models.

2.4.4 Latent Diffusion Models

Despite the impressive results in (un-)conditional image generation, diffusion models remained computationally expensive to train and sample (Rombach et al., 2021). As

explained before, different weightings of the ELBO can shift the learning focus from pixel-based details to more broad features (Diederik P. Kingma and Gao, 2024). However, each operation in the latent space remained expensive due to high-dimensional latents $\boldsymbol{z}_t \in \mathcal{Z} = \mathcal{X} = \mathbb{R}^{H\times W\times 3}$ where typically $H = W \geq 256$. To improve complexity requirements, Rombach et al. (2021) proposed to use a compressed latent space $\boldsymbol{z} \in \mathcal{Z} = \mathbb{R}^{h\times w\times c}$ with dimensions $h < H, w < W$ (specifically $h = w = 64$ for SD) and channels c. The latent dimensions were chosen according to a compression factor of $f = W/w = H/h = 2^n$ for some $n \in \mathbb{N}$. This was achieved by using a VAE-type architecture such that $\tilde{\boldsymbol{x}} = \mathcal{D}(\boldsymbol{z}) = \mathcal{D}(\mathcal{E}(\boldsymbol{x}))$ for a sample $\boldsymbol{x} \sim p(\boldsymbol{x})$ and a jointly trained encoder $\mathcal{E} : \mathcal{X} \mapsto \mathcal{Z}$ and decoder $\mathcal{D} : \mathcal{Z} \mapsto \mathcal{X}$. This concept was termed *perceptual compression*, since it was assumed, by means of the manifold hypothesis (Heaton, 2017, §5.11.3), that one or a few concentrated subspaces of the pixel space $\mathcal{X}$ are suitable for modeling key semantic information of images. The choice of downsampling factor was considered a trade-off between the amount of compression that has to be learned by the diffusion model versus the amount of *first-stage compression* applied by the initial encoding. Regarding the latter point, Rombach et al. (2021) observed that overly compressed latents inevitably lead to a limit on the achievable resulting image quality, regardless of training time.

The full objective of the autoencoding model $(\mathcal{E}, \mathcal{D})$ is a combination of the typical VAE loss (Equation (15)) and an adversarial objective inspired by the GAN framework (Goodfellow et al., 2014). The former consists of a reconstruction loss:

$$\mathcal{L}_{rec}(\boldsymbol{x}, \mathcal{D}(\mathcal{E}(\boldsymbol{x}))) \coloneqq \|\boldsymbol{x} - \mathcal{D}(\mathcal{E}(\boldsymbol{x}))\|_2^2$$

as the mean-square error between the sample and its reconstruction as well as a small regularization term:

$$\mathcal{L}_{reg}(\boldsymbol{x}, \mathcal{E}) \coloneqq \mathrm{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\boldsymbol{z}; \mathcal{E}_\mu, \mathcal{E}_{\sigma^2}\right) \parallel \mathcal{N}\left(\boldsymbol{z}; \boldsymbol{0}, \boldsymbol{I}\right)\right)$$

to reduce the variances of the learned latent space with mean $\mathcal{E}_\mu$ and variance $\mathcal{E}_{\sigma^2}$ of the encoder. The latter contributes the adversarial loss:

$$\underbrace{-\log\left(D_{\boldsymbol{\psi}}\left(\mathcal{D}(\mathcal{E}(\boldsymbol{x}))\right)\right)}_{\coloneqq \mathcal{L}_{adv}(\mathcal{D}(\mathcal{E}(\boldsymbol{x})))} + \log\left(D_{\boldsymbol{\psi}}(\boldsymbol{x})\right)$$

where $D_{\boldsymbol{\psi}}$ is a learned discriminator with parameters $\boldsymbol{\psi}$ that should employ:

$$D_{\boldsymbol{\psi}}(\boldsymbol{x}) \coloneqq \begin{cases} 1, & \text{if } \boldsymbol{x} \sim p(\boldsymbol{x}) \\ 0, & \text{else} \end{cases}$$

to differentiate real data $\boldsymbol{x} \sim p(\boldsymbol{x})$ from 'fake', reconstructed datapoints $\tilde{\boldsymbol{x}} = \mathcal{D}(\mathcal{E}(\boldsymbol{x}))$. The overall training objective is therefore:

$$\mathcal{L}_{\text{Autoencoder}} \coloneqq \min_{\mathcal{E},\mathcal{D}} \max_{\boldsymbol{\psi}} \mathcal{L}_{rec}(\boldsymbol{x}, \mathcal{D}(\mathcal{E}(\boldsymbol{x})) - \mathcal{L}_{adv}\left(\mathcal{D}(\mathcal{E}(\boldsymbol{x}))\right) + \log D_{\boldsymbol{\psi}}(\boldsymbol{x}) + \mathcal{L}_{reg}\left(\boldsymbol{x}, \mathcal{E}\right) \quad (56)$$

The final architecture, termed *Latent Diffusion Model (LDM)*, uses Objective (56) to train the autoencoders together with the following slightly modified version of the simplified diffusion denoising loss (40):

$$\mathcal{L}_{\text{LDM}}(\boldsymbol{\theta}) \coloneqq \mathbb{E}_{(\boldsymbol{x},c)\in\mathcal{D},\boldsymbol{\epsilon}\sim\mathcal{N}(\boldsymbol{0},\boldsymbol{I}),t\sim\mathcal{U}\{1,T\}}\left[\|\boldsymbol{\epsilon} - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c)\|_2^2\right] \quad (57)$$

where $\boldsymbol{z}_t = \sqrt{\alpha_t}\mathcal{E}(\boldsymbol{x}) + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}$ according to Equation (21) for some uniformly sampled timestep $t \in [1, T]$. Rombach et al. (2021) used this architecture together with a few more architectural improvements, such as more flexible conditioning, to train the first large, open-source *Text-to-Image (TTI)* diffusion model, termed *Stable Diffusion (SD)*. As is evident from Objective (57), SD is trained as a noise-prediction diffusion model using a cosine variance schedule $\beta_{[1:t]}$. Additionally, it was one of the first large-scale models, employing a total of 1.45 billion trainable parameters (Rombach et al., 2021). The training data was a diverse collection of image-caption pairs from the LAION-400M dataset (Birhane et al., 2021), which in turn is a curated collection of links to images from the large, open-web CommonCrawl (CommonCrawl, 2024).

2.4.5 Additional Pipeline Components

It remains to explore the actual deep neural network architecture $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)$ used to predict, in the case of SD, the forward process' noise. Commonly, either a UNet (Ronneberger et al., 2015) or a transformer (Vaswani et al., 2017; Peebles and Xie, 2023) type architecture is used. While the latter has recently gained traction in diffusion model contexts, e.g., as the backbone of the newest SD version (Esser et al., 2024), we focus on the long-time standard choice of a UNet architecture. Rombach et al. (2021) extended the standard architecture to allow for more flexible conditioning methods. Formally, $\boldsymbol{\tau}_{\boldsymbol{\theta}}(c) \in \mathbb{R}^{M \times d_\tau}$ represents a domain-specific encoder of external modalities c (e.g., text prompts or images). The actual conditioning is achieved using a *cross-attention* mechanism (Vaswani et al., 2017) such that:

$$\text{Attention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \coloneqq \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{d}}\boldsymbol{V}\right) \text{ with}$$
$$\boldsymbol{Q} \coloneqq \boldsymbol{W}_Q^{(i)}\boldsymbol{\varphi}_i(\boldsymbol{z}_t), \;\; \boldsymbol{K} \coloneqq \boldsymbol{W}_K^{(i)}\boldsymbol{\tau}_{\boldsymbol{\theta}}(c), \;\; \boldsymbol{V} \coloneqq \boldsymbol{W}_V^{(i)}\boldsymbol{\tau}_{\boldsymbol{\theta}}(c)$$

where $\boldsymbol{W}_V^{(i)} \in \mathbb{R}^{d \times d_e^i}$, $\boldsymbol{W}_Q^{(i)} \in \mathbb{R}^{d \times d_\tau}$, $\boldsymbol{W}_K^{(i)} \in \mathbb{R}^{d \times d_\tau}$ are learnable projection matrices and $\boldsymbol{\varphi}_i(\boldsymbol{z}_t) \in \mathbb{R}^{N \times d_e^i}$ a flattened intermediate representation of the UNet that is supposed to be influenced by the external modality. In general, the UNet is designed to encode meaningful information about latents by downsampling and subsequent upsampling of latents in multiple similar blocks of convolutions, a ReLU non-linearity and max-pooling, possibly modified by addition of the cross-attention, softmax vector. Each downsampling step doubles the number of channels and corresponding up- and downsampling blocks are interleaved with residual skip connections (He et al., 2016).

One consequence of the architectural changes made by Rombach et al. (2021) was the possibility to model complex text inputs through cross-attention and text-based domain experts for $\tau_{\boldsymbol{\theta}}$, denoted as text encoders. The initial model used a BERT-tokenizer (Devlin et al., 2019) as a text encoder to infer latent codes of prompts for the cross-attention mechanism. This was later switched (Podell et al., 2024) to the multi-modal CLIP model (Radford et al., 2021) for all versions of Stable Diffusion (Podell et al., 2024; Esser et al., 2024). Its pre-training on a large corpus of image-text pairs was hypothesized to yield more expressive text encodings and thus better learning signals through cross-attention (Podell et al., 2024).

2.4.6 Improved Sampling Methods

Every sampling method for diffusion models has to approximate latents in a discrete fashion, whether it is through SGLD in the NCSN and DDPM frameworks or discrete solvers of the continuous SDE formulation (see Section 2.4.2). However, these can differ greatly in the number of steps required to obtain results of a certain quality. For example, the original DDPM framework used $T = 1000$ (Ho et al., 2020) and the improved version even $T = 4000$ timesteps (Dhariwal and Nichol, 2021b). By formulation of the optimization problem as a Markov chain (see Equation (27)), the reverse process has to iteratively denoise for the same number of steps which becomes very expensive quickly for larger UNets (or generally for more evaluations of $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$). This is also due to the sequential nature of the SGLD sampling in Equation (50), therefore also denoted as *annealed Langevin dynamics* (Y. Song and Ermon, 2019). To overcome this problem, J. Song et al. (2021) propose a modified version of SGLD given as:

$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}{\sqrt{\alpha_t}} \right)}_{\text{'predicted' } \boldsymbol{x}_0} + \underbrace{\sqrt{1-\alpha_{t-1}-\sigma^2(t)}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{x}_t, t)}_{\text{direction pointing to } \boldsymbol{x}_t} + \sigma(t)\boldsymbol{\epsilon} \tag{58}$$

with $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. The authors defined a standard deviation of:

$$\sigma(t) = \eta \sqrt{\frac{1-\alpha_{t-1}}{1-\alpha_t}\beta_t} \tag{59}$$

as a scaled version of the forward process variance from Equation (29) with $\eta \in [0, 1]$. $\eta = 1$ yields the stochastic DDPM sampling while $\eta = 0 \leftrightarrow \sigma(t) = 0$ corresponds to totally deterministic sampling. The latter was thus coined as *denoising diffusion implicit model (DDIM)* and $\eta \in (0, 1)$ is essentially an interpolation between DDIM and DDPM. For DDIM, the process is no longer a Markov chain nor stochastic and the direction towards $\boldsymbol{x}_t$ as well as the predicted sample $\boldsymbol{x}_0$ (see Equation (58)) are only influenced by the inherent noise of the sample itself. In the extreme of $\sigma(t) \to 0$, $\boldsymbol{x}_{t-1}$ becomes a fixed function of $\boldsymbol{x}_0$ and $\boldsymbol{x}_t$ (except for the special case of $t = 1$). Furthermore, J. Song et al. (2021) show that DDIM sampling for any $\eta \in [0, 1]$ can be applied to models trained under the DDPM framework as both training objectives differ only by a constant. Both these facts lead to the assumption that sampling only a subset of latents:

$$\{\boldsymbol{x}_{\tau_1}, \ldots, \boldsymbol{x}_{\tau_S}\} \subset \boldsymbol{x}_{[1:T]}$$

where $\tau \coloneqq [\tau_1, \ldots, \tau_S]$ is an increasing sequence of $[1, \ldots, T]$ with $|\tau| = S$, is a valid strategy to trade-off quality and inference speed. Indeed, J. Song et al. (2021) empirically proved that sampling under $\eta = 0$ using Equations (58) and (59) yielded very comparable results as with $\eta = 1$ however using only every tenth timestep - even though the model was trained under the DDPM Objective (40).

One of the state-of-the-art diffusion model samplers used throughout this work was developed by L. Liu et al. (2022). For a detailed description, we refer to their work which employs pseudo numerical methods to solve an ODE derived from the SDE formulation (see Section 2.4.2), for example using a Runge-Kutta fourth order ODE solver (L. Liu

et al., 2022). Therefore, the method was termed *pseudo numerical methods for diffusion models (PNDM)*. Using this numerical approach, gradients are approximated much more efficiently, yielding similar quality images for $T = 50$ timesteps as compared to $T = 1000$ under DDIM sampling, constituting a 20x performance gain (L. Liu et al., 2022). In essence, PNDM leverages scores from previous timesteps to get a better linear estimate of the current score using standard numerical approximation methods as compared to DDIM which only makes use of the current score. Therefore, it has become a very common choice in combination with SD, which will be used throughout this work.

3 Fairness in Diffusion Models

This section explores recent findings regarding biases in diffusion models and examines various debiasing approaches. It also introduces the key metric used throughout this work to evaluate fairness in generative models.

3.1 Social Biases in Generative Models

Despite their remarkable generative capabilities, diffusion models exhibit notable drawbacks, particularly in their disposition to learn biased data distributions (S. Luccioni et al., 2024). While Dhariwal and Nichol (2021a) demonstrated the superiority of diffusion models among current Text-to-Image (TTI) approaches in terms of image quality and diversity, Perera and Patel (2023) empirically showed that these models consistently display biases towards certain attributes present in the training data. The strength and distribution of preferred attributes heavily depend on the dataset size, although few clear tendencies or rules have been observed (Perera and Patel, 2023). Interestingly, while GANs (Goodfellow et al., 2014) tend to follow the attribute distributions present in their training data (Maluleke et al., 2022; Perera and Patel, 2023), diffusion models show biased attribute distributions even for large, balanced training datasets (Perera and Patel, 2023). Generally, biases can manifest for any combination of attributes. However, as TTI systems gain mainstream usage, considerable focus has been placed on social or human-related biases such as gender, race, ethnicity, age, and geo-culture (Fraser et al., 2023; Mehrabi et al., 2021; Younghyun Kim et al., 2023; Perera and Patel, 2023; S. Luccioni et al., 2024; Vahdat et al., 2021). The worldview encoded in these models is significantly influenced by the composition of their training datasets. For instance, consider all of Stable Diffusion's training datasets, subsumed under the open-source, web-scraped image-text collection LAION-5B (Schuhmann et al., 2022), which predominantly represents Western, male, and younger demographics from developed countries (Perera and Patel, 2023; De Simone et al., 2023; A. Luccioni and Viviano, 2021; Worldbank, 2024). Prior to model training, the data is filtered using the CLIP model, which has been observed to be more accurate and reliable in identifying Caucasian-appearing people (Wolfe and Caliskan, 2022). Both the dataset itself and the filtering process may inadvertently introduce additional biases, particularly in the representation of minority groups (Naik and Nushi, 2023).

S. Luccioni et al. (2024) validated these biases by generating images of persons in different professions or occupations in combination with various social traits like gender or race using SD. A clustering analysis revealed a bias towards whiteness and masculinity for the average generated image. In general, outputs mostly follow the US Bureau of Labor Statistics (BLS, 2024) regarding gender, race, and age distributions but consistently amplify biases and under-represent marginalized identities (S. Luccioni et al., 2024; Bianchi et al., 2023). Naik and Nushi (2023) extended this line of research by incorporating a novel inspection of location or geographical biases in such systems. Similar to prior research, the authors showed that a majority of the models' generated outputs are interpreted to be American or German locations. Generally, most contemporary diffusion models inherit or even exaggerate typical Western stereotypes, which can be mostly attributed to the prevalence and availability of such data in the training sets (Perera and Patel, 2023).

3.2 Bias Mitigation Techniques

This section briefly summarizes state-of-the-art approaches for mitigating biases in diffusion models. Generally, most methods focus on pipeline components related to prompting abilities (X. Shen et al., 2024). We broadly categorize existing methods into two categories: those that act in the training stage and those that perform debiasing during deployment, i.e., modifying the generative process. The former domain focuses on different modifications of the existing loss function or subsequent finetuning of parameters using a new, custom-designed loss. Yeongmin Kim et al. (2024) propose a principled change to the simple denoising loss (40). A time-dependent re-weighting of the ELBO is implemented, based on the density ratio of estimated and target distribution. This is supposed to yield fair models after training even on highly biased datasets. Esposito et al. (2023) present a straightforward method of finetuning the UNet of the SD model and its more recent successor, *Stable Diffusion XL (SDXL)* (Podell et al., 2024), using synthetically generated data from the SDXL model. While improvements for SD were noticeable, those for SDXL were rather mixed with only subtle improvements. On the other hand, X. Shen et al. (2024) have shown that finetuning the text encoder instead of UNet yields slightly worse fairness but results of clearly superior quality as opposed to the UNet. Refer to Section 5.1.4 for an elaborate explanation of their custom loss function designed to finetune the model for fairness and semantics preservation. A hybrid between training and deployment time methods is provided by J. Li et al. (2024). The authors train an additional small linear network on top of the frozen text encoders to generate debiased prompt embeddings. It is trained by a combination of fairness and semantics preservation loss and implemented as a layer on top of the text encoder during inference. While the debiasing was effective, especially on unseen prompts, it requires notable resources and scales unfavorably to more attributes. Following a similar idea, Bansal et al. (2022) analyzed the rather simple linguistic approach of appending ethical interventions to prompts for debiasing purposes, such as *"if all individuals can be a <occupation> irrespective of their gender"* for some occupation. However, efficacy was limited. E. Kim et al. (2023) propose to instead finetune an additional soft token, inserted before a specific prompt, with the finetuning loss targeted towards fairness. However, applicability was limited as tokens were finetuned on a per occupation basis. Other approaches include projecting out biased directions in the text embeddings themselves (Chuang et al., 2023) or the score predictions (Z. Wang et al., 2023). Another promising avenue seems to be a modification or finetuning of the cross-attentions weights used for text conditioning purposes (see Section 2.4.3) (Orgad et al., 2023; Gandikota et al., 2024). Lastly, there are inference time approaches such as Fair Diffusion (Friedrich et al., 2023). It uses a technical method to steer the generative process along semantic directions of textual descriptions without external dependencies. Refer to Section 5.1.2 for a detailed description. Another approach by Olmos et al. (2024) employs a lightweight SVM (Cortes and Vapnik, 1995) that was trained to classify sensitive attributes from latents. It is subsequently used during deployment to guide latents for fairness purposes. Parihar et al. (2024) similarly leverages lightweight classifiers of a newly discovered latent space of diffusion models (see Section 4.3) to guide towards fairer results using classifier guidance (see Section 2.4.3).

3.3 Fairness Definitions for Generative Models

At its core, fairness in generative models aims to ensure that generated outputs do not exhibit or amplify biases related to sensitive attributes. To quantify this, we define a suitable fairness metric. Consider a batch of images $\boldsymbol{x}^{[1]}$ and a sensitive attribute $S = s_1, \ldots, s_K \in \mathcal{S}$. For example, S could represent gender with classes $s_1 =$ *female"* and $s_2 =$ *male"*. It is common practice to employ a high-quality image classifier $\mathcal{C} : \mathcal{X} \mapsto [0,1]^K$ to obtain class probabilities for each datapoint automatically. Crucially, assessing the classifier's own biases regarding the sensitive attributes is essential to ensure reliable results.

Most authors define a generative model as fair if the probability of a generated sample being classified into any of the sensitive attribute classes is equal (X. Shen et al., 2024; Friedrich et al., 2023; Parihar et al., 2024). However, as highlighted by De Simone et al. (2023), this approach may not fully capture the flexibility and widespread usage of generative models. Different stakeholders and use cases may require the model to adhere to various attribute distributions. Therefore, we consider fairness as a more general distributional alignment problem between a discrete target distribution $\boldsymbol{p}\text{tar} \in [0,1]^K$ for some sensitive attribute S with K classes, and the model's distribution $\boldsymbol{p\theta}$ over the same classes. A suitable and commonly used metric in this setup is the *Fairness Discrepancy (FD)* (Parihar et al., 2024; K. Choi et al., 2020), defined as:

$$\mathrm{FD}(\boldsymbol{p}_{\boldsymbol{\theta}}, \boldsymbol{p}_{\text{tar}}) = \left\| \boldsymbol{p}_{\text{tar}} - \mathbb{E}_{\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})}[\mathcal{C}(\boldsymbol{x})] \right\|_2^2 \tag{60}$$

The expectation can be computed as a Monte Carlo estimate by creating a sufficiently large dataset of generated datapoints to approximate the model's inherent distribution. This metric provides a flexible and interpretable measure of how well a generative model aligns with desired fairness criteria, allowing for customization based on specific use cases and ethical considerations.

We seek to elucidate the boundaries of this metric under different conditions. While the lowest possible value is always 0, the upper boundary varies depending on the number of classes and the target distribution. For a balanced target distribution over K classes $\boldsymbol{p}_{\text{tar}} = [1/K, \ldots, 1/K]$, the highest bias is achieved if all predicted probability mass is focused on one class:

$$\mathbb{E}_{\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})}[\mathcal{C}(\boldsymbol{x})] = [1, 0, \ldots, 0]$$

assuming, without loss of generality, that the mass is concentrated on the first class. The difference vector is $\boldsymbol{d} = [1 - 1/K, -1/K, \ldots, -1/K]$, leading to:

$$\|\boldsymbol{d}\|_2^2 = \left(1 - \frac{1}{K}\right)^2 + (K-1)\left(\frac{1}{K}\right)^2 = 1 - \frac{1}{K} \tag{61}$$

Thus, a binary attribute yields bias values in the range $[0, 0.5]$. Due to its relevance for Section 5, we also consider a binary attribute with target distribution $\boldsymbol{p}\text{tar} = [0.75, 0.25]$. The maximum bias is obtained when:

$$\mathbb{E}_{\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x})}[\mathcal{C}(\boldsymbol{x})] = [0, 1]$$

yielding $\boldsymbol{d} = [-0.75, 0.75]$, and the upper bound is $\|\boldsymbol{d}\|_2^2 = 1.125$.

4 Mitigating Biases in Diffusion Models

Building upon the theoretical foundations of diffusion models and the issues of bias in generative models discussed in previous sections, we now introduce a novel approach to mitigate these biases. This section presents *Debias Diffusion (DD)*, a method designed to reduce biases related to sensitive attributes such as gender, race, and age across contemporary TTI diffusion models. Debias Diffusion leverages both syntactic information from input prompts and semantic content from the models' latent spaces to flexibly adjust to any given fairness criteria and usage contexts. In the following subsections, we outline the motivation behind this approach, elucidate its theoretical underpinnings, and provide a comprehensive explanation of the algorithm. We begin by examining the challenges in debiasing generative models and the rationale for the developed methodology. Subsequently, we explore how different stages of image generation can be strategically manipulated to achieve fairer outputs.

4.1 Motivation

The development of Debias Diffusion was driven by the imperative to address three principal challenges in debiasing diffusion models: complexity, performance, and usability. These factors guide the decision-making process regarding the presented algorithm in Section 4.4 and the chosen evaluation metrics in Section 5.3.

Complexity Complexity is a notable aspect in debiasing efforts. Existing methods typically fall into two categories: training-time approaches (Teo et al., 2023; K. Choi et al., 2020; Xu et al., 2018; X. Shen et al., 2024; Esposito et al., 2023) that modify model parameters, and inference-time techniques (Y. Choi et al., 2024; J. Li et al., 2024; Friedrich et al., 2023) that alter the generative process on the fly. Training-time approaches, while potentially effective, often demand substantial computational resources and lack flexibility. For instance, the fine-tuning of Stable Diffusion in X. Shen et al. (2024) requires approximately 48 hours on 8 NVIDIA A100 GPUs, despite only training the model's text encoder with a parameter-efficient LORA (Hu et al., 2022) adapter. Conversely, inference-time approaches offer greater adaptability but may struggle with generalization across diverse prompts. Furthermore, they may degrade inference-time performance by introducing new complexities through the use of additional models in the pipeline, such as a classifier (Parihar et al., 2024) or a linear network (J. Li et al., 2024).

Performance The efficacy of a debiasing algorithm is a crucial aspect, denoting the ability to debias a wide range of models in a precise and effective manner. A flexible algorithm would enable the model to match arbitrary probability distributions over a wide range of possible sensitive attributes or combinations thereof. However, in many scenarios, stakeholders may expect the debiasing to work in an isolated fashion to preserve the model's utility and inherent data manifold. For example, X. Shen et al. (2024) employ a face detection model in their fine-tuning loss with an additional pixel-based gradient stop to prevent changes outside the domain of interest (human depiction). Similarly, J. Li et al. (2024) train a set of linear mappings to debias prompt embeddings using

a combination of a fairness penalty loss and a text consistency loss that is supposed to minimize the discrepancy between pre- and post-transformed embeddings. Moreover, performance considerations extend beyond bias mitigation to encompass the preservation or enhancement of the model's original capabilities. This includes maintaining or even improving image quality, semantic coherence, and prompt adherence. Striking the right balance between achieving fairness as well as attaining prescribed qualities is essential, otherwise debiasing efforts may inadvertently compromise the model's overall performance or lack effectiveness.

Usability The complexity criterion focuses on the engineering and implementation challenges of any given algorithm. Additionally, a debiasing algorithm should be flexible and easy to use for the average end-user. Therefore, it is important to note that fairness is an inherently subjective and context-dependent measure. Generative models are being used by a variety of different social groups with different opinions and expectations on the definition and distribution of sensitive attributes. As noted in De Simone et al. (2023), many current debiasing approaches lack the ability to flexibly adjust to different worldviews that a stakeholder may want represented in any given context. These factors can severely degrade the debiased models' usability and therefore limit the widespread adoption of such systems. Consequently, a debiasing approach may greatly benefit from the ability to make ad-hoc adjustments to the sensitive attributes' distribution.

By addressing these three key aspects – complexity, performance, and usability – Debias Diffusion aims to provide a comprehensive solution to the challenge of mitigating biases in diffusion models. The following sections will delve into the technical details of how this approach navigates these challenges, offering a nuanced and effective method for creating fairer generative outputs.

4.2 Prioritizing Pretext Tasks in Diffusion Models

As explained in Section 2, diffusion models learn to restore data corrupted with various noise levels. These so-called *pretext tasks* (J. Choi et al., 2022) provide the model with informative challenges of varying difficulty to refine its feature extraction capabilities. To gain deeper insight into what the diffusion model learns at each noise level, we consider the work of J. Choi et al. (2022). The authors proposed re-weighting the simple denoising diffusion loss (see Equation (40)) to emphasize timesteps close to T during the training process. They hypothesized that pretext tasks with higher noise levels are more challenging, requiring the model to learn domain-specific knowledge due to the lack of recognizable content or patterns. The resulting models empirically demonstrated significantly improved performance. Therefore, following J. Choi et al. (2022), the generative process is divided into three stages with distinct characteristics throughout this work:

1. A coarse stage to generate broad image features (e.g., color scheme, subjects)
2. A content stage in which rich features are employed
3. A clean-up stage to enhance photorealism and quality

Figure 3: VAE decoded image predictions of Stable Diffusion. Colored borders indicate different stages where blue denotes the coarse stage, red the content stage, and green the clean-up stage. The prompt *"A photo of a diplomat"* was employed for guidance with $T = 50$ timesteps.

These phases are separated based on the model's SNR values, with the content phase starting at $\tau_1 = 10^{-2}$ and the clean-up stage at $\tau_2 = 10^{0}$. Each stage is visibly marked in Figures 2b, 3, and 4. To better understand each pretext task, Figure 3 depicts the image predictions of all timesteps of SD's generative process for a test prompt. These are denoted as $\boldsymbol{x}_0^{[1:T]}$ where $\boldsymbol{x}_0^t = \hat{\boldsymbol{x}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c)$ is the image ($\boldsymbol{x}_0$) prediction of the model at timestep t (refer to Equations (35) and (58)). Formally, for a given timestep t, these are obtained as:

$$\boldsymbol{x}_0^t = \mathcal{D}\left(\frac{1}{\sqrt{\alpha_t}}\left(\boldsymbol{z}_t - \sqrt{1-\alpha_t}\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c)\right)\right)$$

where $\boldsymbol{z}_0^t$ is the predicted last latent from any intermediate latent $\boldsymbol{z}_t$ and $\mathcal{D}$ the perceptual decoder. During the coarse stage, where $t \approx T$, the model's predictions are very blurry and miss clearly defined features. In the content stage, the depiction of the subjects is continuously being refined and their appearance as well as composition are determined. Notably, the light-skinned person's features and attributes (such as gender, appearance, etc.) are more stable and refined in earlier timesteps compared to the dark-skinned person. This observation, consistent through many more similar examples, suggests that the model makes stronger early assumptions for attributes it has a bias towards, which will be investigated further in subsequent sections.

To explore the stability of the generative process trajectory at each timestep, a re-diffusion experiment similar to Um et al. (2024) is performed. Given the final image $\boldsymbol{x}_0$ for the aforementioned prompt, it is first perturbed to a noisy latent $\boldsymbol{z}_t \sim q(\boldsymbol{z}_t \mid \boldsymbol{z}_0)$ for all

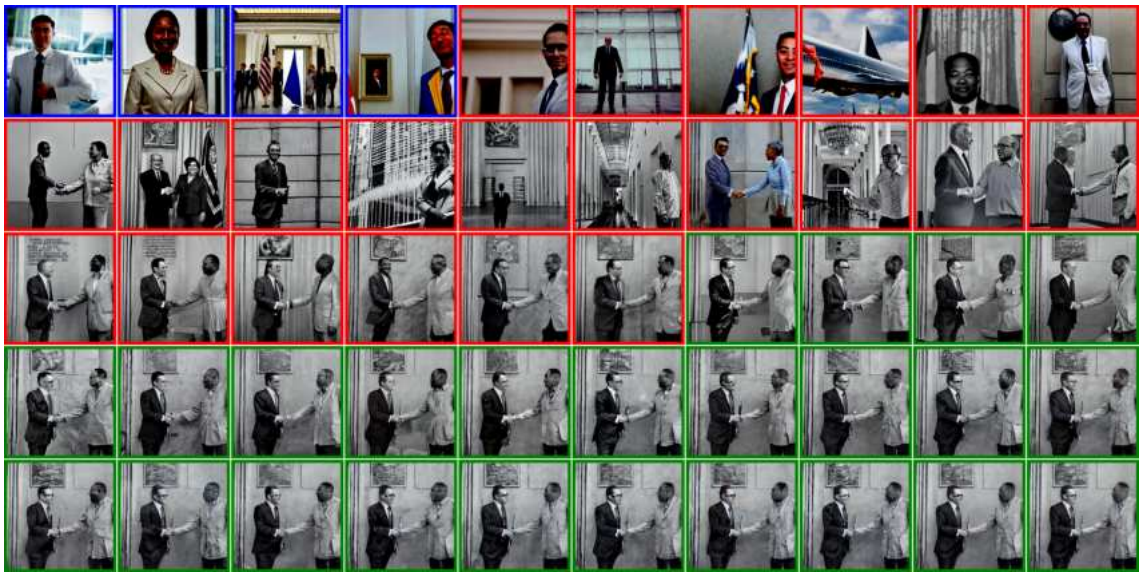


Figure 4: Re-diffused versions of the same image up to different timesteps using the SD model and PNDM solver. Colored borders indicate different stages where blue denotes the coarse stage, red the content stage, and green the clean-up stage. The prompt $c =$ *"A photo of a diplomat"* was employed for guidance.

$t \in [1, T]$ using the forward transitioning kernel eq. (21) and the perceptual encoder $\mathcal{E}$:

$$\begin{aligned} \boldsymbol{z}_0 &= \mathcal{E}(\boldsymbol{x}_0) \\ \boldsymbol{z}_t &= \sqrt{\alpha_t}\boldsymbol{z}_0 + \sqrt{1-\alpha_t}\boldsymbol{\epsilon} \end{aligned} \tag{62}$$

with random noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Assuming the image has been noised up to some timestep $\tau \in [1, T]$, the original image is approximated by iteratively computing the latents until $t = 0$:

$$\begin{aligned} \boldsymbol{z}_{t-1} &= \text{PNDM}(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c)) \;\; \forall t \in [1, \tau] \\ \boldsymbol{x}_0^{\tau} &= \mathcal{D}(\boldsymbol{z}_0). \end{aligned} \tag{63}$$

using the sampler of choice (PNDM in this case)[1].This process yields a set of images $\boldsymbol{x}_0^{[1:T]}$, one for each τ. Figure 4 displays all re-diffused versions for the same seed, generated using SD. Re-diffusion until the coarse stage drastically changes the image composition and aesthetics. As τ is lowered and the content stage is entered, re-diffused images appear much more similar to the original $\boldsymbol{x}_0$, with similar overall image composition and color grading. Re-diffused images from the clean-up stage differ only slightly while maintaining consistent overall composition. Notably, in contrast to the clean-up stage, re-diffused images from the earlier steps of the content stage depict subjects with varying sensitive attributes such as gender or race. These findings support the initial hypothesis and indicate the model's ability to change sensitive attributes in isolation.

[1]Note that PNDM in practice also conditions on previous score estiamtes, which we omit for notational clarity (see Section 2.4.6).

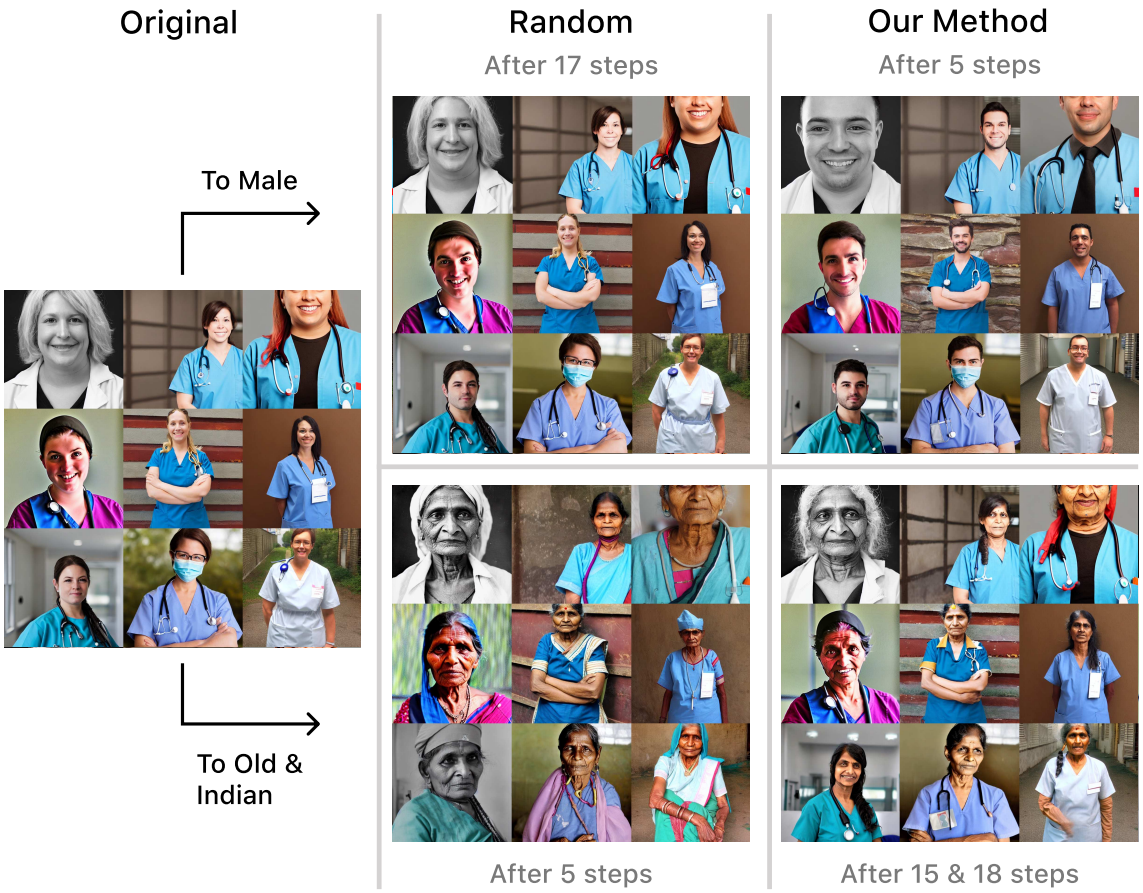


Figure 5: Generations of the prompt *"A photo of a nurse"* using Stable Diffusion. Images at the same positions in the grid were generated using the same initial noise. In the upper row, an attempt is made to depict only male nurses, and in the lower row, the goal is to depict only old Indian female nurses using "random" switching timepoints and estimated timepoints from the proposed method.

Furthermore, empirical testing has shown that diffusion models are capable of depicting much more diverse images compared to their default generations when properly incentivized. For example, Younghyun Kim et al. (2023) reported that the depiction of nurses in Stable Diffusion is almost completely dominated by female-appearing persons. However, explicitly prompting for male nurses results in a wide variety of high-quality images showing male nurses. To validate this hypothesis, we conducted an experiment on this highly biased domain of SD. The model was tasked to generate a batch of images using the prompt $c =$ *"A photo of a nurse"* which mainly resulted in the depiction of young, Caucasian, female persons, with gender bias being especially pronounced. Inspired by Y. Choi et al. (2024), we tried generating images using the same prompt and seed (initial noise $\boldsymbol{z}_T$) that depict only male nurses by changing the prompt to $\hat{c} =$ *"A photo of a male nurse"* at a certain timestep in the generative process. The results are illustrated in Figure 5. First, a random timestep $\tau_{gender} = 33$ was selected in the middle of the content phase (i.e., 17 steps into the generative process) at which the modifier *"male"* was inserted. As evident in the figure, the generations largely remain almost identical, and the model fails to render

the desired attribute's class for almost all examples. In contrast, the proposed method (see Section 4.4) measures the gender bias of the current batch early in the process to improve upon this. As it measures a high value for the gender attribute, the model reverts to the start of the content phase at $\tau_{gender} = 45$ and continues the generation with the modifier inserted. Consequently, the model successfully renders the desired attribute class on every batch example while still retaining the context of the original generations.

To further illustrate the importance of choosing an appropriate insertion point of the modifier, we attempted to generate images of old Indian female nurses using the same batch of initial noise. The results of inserting the modifiers at the timestep evaluated for gender, given as $\tau_{age} \coloneqq \tau_{race} \coloneqq \tau_{gender} = 45$, are displayed in the second row of the second column in Figure 5. The results successfully display old Indian women; however, compared to the original batch of images, most new generations display a notable lack of suitable content regarding the prompt's context (i.e., depictions of nurses). In technical terms, this is the result of an increased discrepancy between the latents' trajectories due to the early introduction of different conditionings. In contrast, the proposed approach detects suitable insertion timesteps for both attributes independently ($\tau_{race} = 35$ and $\tau_{age} = 32$), depending on their respective bias. The results are presented in the adjacent column and same row. As evident, the desired attribute classes are still being rendered successfully while most images retain much more of the original generations' surrounding context, leading to generations that adhere much better to the requested context (i.e., nurses).

These observations support the hypothesis that the strength of the model's bias in a certain domain correlates with its ability to change semantic attributes during the generative process. We also refer to this as the *main hypothesis* in reference to the developed method throughout the rest of this work. The subsequent sections will elaborate on how bias is measured during generation, along with other aspects that automate and optimize the debiasing process.

4.3 Training h-Space Classifiers

Building on the insights from the pretext task analysis 4.2, this part details how to leverage semantic information present in the early stages of the generative process. This information is crucial for making informed decisions about when and how to debias the currently generated batch. While other generative architectures, such as GANs (Goodfellow et al., 2014), produce latents that encode universal and independent semantic directions (Y. Shen et al., 2019), latents produced in (latent) diffusion models contain limited semantic content, especially in the early generation stages (J. Choi et al., 2022). Olmos et al. (2024) demonstrated the possibility of separating intermediate latents at later timesteps with respect to the sensitive attribute gender using a *Support Vector Machine (SVM)* (Cortes and Vapnik, 1995). However, accuracy was limited, particularly for larger models such as SDXL. Another natural approach might be to classify the image predictions $\boldsymbol{x}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c)$ (see Figure 3). However, as demonstrated by Parihar et al. (2024), diffusion models employ a much richer and more consistent latent space. Consider the model's UNet as a

combination of an encoder and decoder:

$$\begin{aligned} \boldsymbol{h}_t &= \boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{ENC}(\boldsymbol{z}_t, t, c) \\ \boldsymbol{z}_{t-1} &= \boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{DEC}(\boldsymbol{h}_t, t, c) \end{aligned} \tag{64}$$

where $\boldsymbol{\epsilon}_{\boldsymbol{\theta}} = \boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{DEC} \circ \boldsymbol{\epsilon}_{\boldsymbol{\theta}}^{ENC}$.[2] The aforementioned semantic latent space is spanned by the outputs of the UNet's encoder $\boldsymbol{h}_t \in \mathcal{H}$ and is denoted as *h-space* (Kwon et al., 2023). These vectors are also called the bottleneck vectors of the UNet since they employ the largest channel and lowest spatial dimension of all intermediate latents. For example, the UNet implementation of SD employs $\boldsymbol{h}_t \in \mathbb{R}^{1280\times 8\times 8}$, which is between the latents $\boldsymbol{z}_t \in \mathbb{R}^{4\times 64\times 64}$ and image $\boldsymbol{x}_0 \in \mathbb{R}^{3\times 512\times 512}$ in terms of raw size. Kwon et al. (2023) first observed that making informed changes to these latent representations after the coarse stages of the generative process can flexibly enforce or suppress certain semantic concepts in the resulting image in isolation. Furthermore, applied modifications for the same concepts were very similar across timesteps with consistent effects across different samples.

Leveraging these h-space representations, Parihar et al. (2024) train an ensemble of lightweight linear classifiers $\eta_t^{(i)} : \mathcal{H}_t \to [0,1]^{K_i}$ over all timesteps for sensitive attribute $S^{(i)}$ with K_i classes. To do so, a pre-labeled dataset

$$\mathcal{D}^{(i)} = \{(\boldsymbol{x}^n, \boldsymbol{y}^n)\}_{n=1}^N$$

of high-quality real images $\boldsymbol{x}^n$ and corresponding labels $\boldsymbol{y}^n \in [0,1]^{k_i}$ for each sensitive attribute $S^{(i)}$ is required. First, each sample is perceptually compressed as $\boldsymbol{z}_0^n = \text{VAE}_{ENC}(\boldsymbol{x}^n)$. Afterwards, all noisy latents $\boldsymbol{z}_{[1:T]}^n$ are obtained using the deterministic forward process of the DDIM sampler with $\eta = 0$ (see Section 2.4.6). Encoding each latent using the Unet's encoder in Equation (64) yields an h-space representation $\boldsymbol{h}_{[1:T]}^n$ of the given samples. Sorting these per timestep for all elements together with the corresponding image's label provides a training dataset

$$\mathcal{D}_t^{(i)} = \{(\boldsymbol{h}_t^n, \boldsymbol{y}^n)\}_{n=1}^N$$

for each classifier $\eta_t^{(i)}$. Note that it is equally possible to use the same dataset for training all classifiers if the dataset provides accordingly labeled datapoints for every attribute.

The present work employs a similar strategy to obtain these classifiers, but without the need for an externally collected and classified dataset, while achieving comparable overall performance. The focus is on a setting that solely requires debiasing human-related subjects, though the method can be similarly applied to other domains. To obtain a proper training dataset concerning sensitive attribute $S^{(i)}$, a certain prompt template is employed to incentivize the model to create subjects with a desired class. An occupation $o \in \mathcal{O}$ is randomly sampled from a pre-defined list $\mathcal{O}$ as well as a sensitive class $s \in S^{(i)}$ with equal probability and included in a prompt $c =$ *"A photo of the face of a <s><o>"*. Subsequently generating an image using this prompt should yield a sample faithful to the prescribed class. Therefore, collecting the accordingly generated h-space latents $\boldsymbol{h}_{[1:T]}$ paired with a one-hot label $\boldsymbol{y}$ derived from the inserted class s provides a training sample

[2] Note that some contemporary UNet implementations may employ an additional middle block. Following the source implementation from Kwon et al. (2023), we propose to use the output of the middle block.

for each classifier $\eta_t^{(i)}$. Repeating this process N times, preferably efficiently in batches, yields a labeled dataset

$$\mathcal{D}_t^{(i)} = \{(\boldsymbol{h}_t^n, \boldsymbol{y}^n)\}_{n=1}^N$$

to enable the training of the classifiers. We term this approach *self-labeled h-space classification*, as it eliminates the need for an externally labeled dataset of existing images.

However, as noted before and demonstrated by Rassin et al. (2024), depending on the subject and attributes to display, TTI diffusion models may occasionally fail to render the desired attribute or subject in the final generation. This can cause unrelated or incorrectly labeled training samples which would degrade the resulting classifiers' performance. Refer to Rassin et al. (2024) for an empirical ablation on this phenomenon. Hence, it is beneficial to use additional support from a face detection model $d_{face} : \mathcal{X} \mapsto \{0, 1\}$ as well as a high-quality image classifier $\mathcal{C}^{(i)} : \mathcal{X} \mapsto [0, 1]^{k_i}$ for each attribute $S^{(i)}$. Generation of training samples is performed as before, however only images with detected faces are included and the classifier's probabilities are used as training labels. Note that the desired attributes are still inserted into the prompt to obtain more data from minority groups where an overall balanced dataset is the goal. Formally, the training data for each timestep is obtained as:

$$\mathcal{D}_t^{(i)} = \{(\boldsymbol{h}_t^n, \boldsymbol{y}^n) \mid d_{face}(\boldsymbol{x}^n) = 1,\ \boldsymbol{y}^n = \mathcal{C}^{(i)}(\boldsymbol{x}^n)\}_{n=1}^N.$$

To enforce a balanced dataset, samples classified as sensitive class s are rejected after collecting the maximum of K/N samples of it. Additionally, the class is excluded from the set of modifiers inserted into the prompt template.

Depending on the scenario, it may be more convenient to use a pre-labeled dataset, a combination of detection and/or classification models, or no other pre-requisites at all. The presented three methods perform comparably well for the sensitive attributes under consideration (gender, age, and race), as shown in Section **??**, while arguably allowing for more flexibility at roughly the same computational cost and very similar classification accuracy. Note that we found it beneficial to create one training dataset for all three sensitive attributes simultaneously. However, the number of attributes that can be generated simultaneously is limited by the model's generative and prompt adherence capabilities as well as the type of subject and attributes to render.

4.4 Debias Diffusion

This section presents the developed method, termed *Debias Diffusion*. To provide context, let $\boldsymbol{\theta}$ be a pretrained diffusion model with a corresponding UNet $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ that was trained on biased data with respect to some sensitive attributes $S^{(1:M)}$. We explicitly note that the model may have been trained to perform either score, image mean or noise prediction while we assume the latter one in the following (see also Sections 2.4.1 and 2.4.2). The attributes are assumed to be independent, i.e., having one class in one attribute during generation does not suppress or enforce other attributes' classes. We will discuss this assumption in Section 5.5. A prompt c is used to generate a batch of images $\boldsymbol{x}^{[1:N]}$ which may be debiased regarding a sensitive attribute $S^{(i)}$. The approach seeks to align the generated discrete attribute distribution $\boldsymbol{p}_{\boldsymbol{\theta}}^{(i)}$ with the stakeholder's desired discrete target distribution $\boldsymbol{p}_{tar}^{(i)}$ over the sensitive attribute $S^{(i)}$. Generally, images may be generated

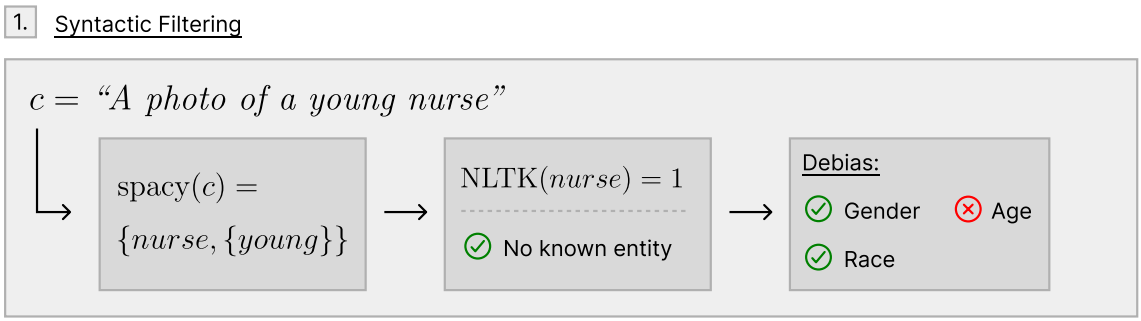


Figure 6: Schematic overview of the syntactic filtering process in Debias Diffusion to filter human-related nouns in prompts.

one by one in a subsequent manner or as batches in parallel. The algorithm focuses on the latter use-case as it is computationally more efficient. However, the approach is similarly applicable to subsequent generations. The approach is conceptually divided into three phases, explained hereafter: Syntactic Filtering, h-Space Bias Estimation, and Bias-Sensitive Attribute Insertion.

4.4.1 Syntactic Filtering

Most current TTI diffusion models aim to create diverse content based on complex prompts. Therefore, it is essential to first detect if the subject to be debiased is or will be present in the generation, an aspect often not explicitly handled by adjacent works (Friedrich et al., 2023; Y. Choi et al., 2024; Parihar et al., 2024). Modifying a model's outputs on unrelated domains may cause undesirable side-effects. To address this, the prompt's syntactic structure is analyzed prior to the actual generative process. As noted by Rassin et al. (2024), this can be achieved reliably and efficiently using the *spaCy natural language processing (NLP) toolkit* (Honnibal et al., 2020). At its core, spaCy employs a dependency parsing algorithm to construct a hierarchical representation of syntactic relationships within a given text. The dependency parser in spaCy uses a transformer-based architecture (Vaswani et al., 2017), which has demonstrated state-of-the-art performance in various NLP tasks (Honnibal et al., 2020). This implementation utilizes a newer version which employs a pre-trained BERT (Devlin et al., 2019) model. The parser generates a directed acyclic graph $G = (V, E)$, where V represents the set of tokens in the input prompt, and E represents the set of directed edges denoting syntactic dependencies between tokens. Each edge $e \in E$ is labeled with a dependency relation r from a predefined set of grammatical relationships $\mathcal{R}$. The algorithm traverses G to identify all entity nouns $n \in \mathcal{N} \subset V$ that are not direct modifiers of other nouns. These entity nouns typically correspond to the primary objects or concepts in the generated image (Rassin et al., 2024). For each identified noun n, the algorithm recursively collects its modifiers $m \in \mathcal{M} \subset V$, where $(m, n) \in E$ and the edge label r belongs to a subset of modifier relations $R_{mod} \subset \mathcal{R}$. The set R_{mod} includes key syntactic relations that capture relevant aspects of noun modification for the debiasing approach:

1. Adjectival modification (amod): e.g., *"the young doctor"*

2. Nominal modification (nmod): e.g., *"the scientist of Asian descent"*
3. Compound nouns (compound): e.g., *"the African American engineer"*
4. Adverbial modifiers (npadvmod): e.g., *"An exceptionally talented female CEO"*

These specific modifier types are focused on as they are most pertinent to capturing attributes relevant to potential biases in image generation. For instance, given the prompt $c =$ *"A young female programmer working on a complex algorithm"*, the method would identify *"programmer"* as the primary entity-noun, with *"young"* and *"female"* as its adjectival modifiers. The phrase *"working on a complex algorithm"* would be captured as additional context but not directly relevant to the debiasing strategy. By systematically analyzing prompts in this manner, a structured representation of each prompt is obtained. Formally, for each prompt c, we define a function spacy(c) that yields a set of tuples

$$\text{spacy(c)} = \{(n_j, M_j)\}_{j=1}^{J} \tag{65}$$

where $n_j \in \mathcal{N}$ represents an entity-noun and $M_j \subset \mathcal{M}$ is the set of its associated modifiers. Each modifier $m \in M_j$ is paired with its corresponding syntactic relation $r \in R_{mod}$.

Subsequently, this representation should be analyzed to infer which nouns specifically relate to human entities. As empirically shown in Sections 5 and 6.1.1, this can be achieved reliably by leveraging the *Natural Language Toolkit (NLTK)* (Bird et al., 2009), specifically its *WordNet* interface. WordNet is a lexical database that organizes words into semantic networks, providing rich hierarchical relationships between concepts. Of particular interest for the given task are the hypernym relationships, which represent "is-a" connections between more specific and general concepts. Formally, a function NLTK : $\mathcal{N} \to \{0, 1\}$ is defined that maps each noun n from the extracted set of nouns $\mathcal{N}$ to a binary value indicating whether it describes a human entity:

$$\text{NLTK}(n) \coloneqq \begin{cases} 1, & \text{if } \exists s \in \text{Syn}(n), t \in T(s) : \textit{"person"} \in t \vee \textit{"human"} \in t \\ 0, & \text{else} \end{cases}$$

Here, Syn(n) represents the set of synsets (synonym sets) for noun n and $T(s)$ denotes the transitive closure of hypernyms for synset s. The transitive closure refers to the complete set of hypernyms reached by recursively following the hypernym relationships, essentially capturing all ancestral hierarchy for each synset of the given noun, checking if any hypernym contains the terms *"person"* or *"human"*. Applying this function to the structured representation obtained from spaCy in Equation (65) yields a refined set of human-related noun-modifier pairs:

$$H \coloneqq \{(n_j, M_j) \mid (n_j, M_j) \in \text{spacy}(c),\ \text{NLTK}(n) = 1\}$$

for any given prompt c. This set contains only those noun-modifier pairs where the noun has been identified as human-describing. The approach captures not only direct references to people but also occupations, roles, and other nouns that may indirectly describe human entities. For instance, terms like *"doctor"*, *"artist"*, or *"athlete"* would be correctly identified as human-describing, even though they don't directly contain the words *"person"* or *"human"*.

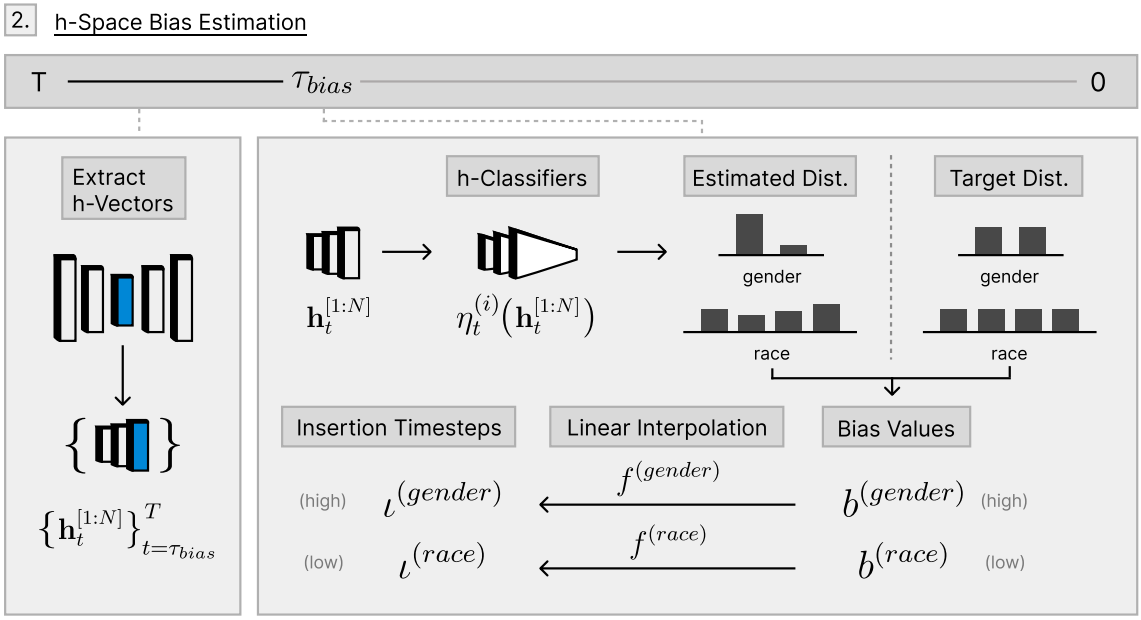


Figure 7: Schematic overview of the bias estimation process in Debias Diffusion using h-space classifiers to determine suitable insertion timesteps for each attribute's desired class.

To further refine the approach and avoid inadvertently modifying the appearance of known entities, NLTK's named entity recognition (NER) capabilities are leveraged. Specifically, the Stanford NER tagger (Finkel et al., 2006) integrated within NLTK is utilized, which employs a machine learning-based approach to identify and classify named entities in text. By excluding nouns classified as named entities (such as specific people or organizations) from the set of detected human-describing nouns, the debiasing process respects the integrity of explicitly named individuals or groups. This additional step enhances the utility of the method (see Section 4.1), allowing for targeted debiasing while preserving the intended representation of known entities within the generated images. It is implicitly assumed that a noun n with $\text{NLTK}(n) = 1$ is not a known entity.

The algorithm subsequently only intervenes with the generative process for the prompt c if $H(c) \neq \emptyset$, meaning that at least one human-related noun was detected. If so, a random noun-modifier pair $(n, M) \in H(c)$ is selected. The attributes to debias are then further filtered by checking the modifiers $m \in M$ present. If $\exists m \in M, s \in S^{(i)} : m = s$, meaning that the user explicitly specified the desired attribute's class, this specification is respected and the algorithm refrains from modifying that particular attribute. Otherwise, the debiasing procedure continues as explained in the next sections. The whole process is schematically illustrated in Figure 6. In this schematic example, the sensitive attribute age would be excluded from any debiasing purposes since it was explicitly specified in the prompt as detected by spaCy.

4.4.2 h-Space Bias Estimation

The Debias Diffusion algorithm leverages h-space classifiers (see Section 4.3) to evaluate the bias of the generated batch independently for each sensitive attribute. As elucidated in Section 4.2, the main hypothesis suggests that some attributes may require earlier modification in the generative process than others to successfully alter the desired attributes, given that the latents' trajectory is biased towards certain default paths. To quantify this intuition, Debias Diffusion employs the Fairness Discrepancy (FD) metric (see Section 3.3) which measures the bias of the currently generated batch at a certain timestep τ_{bias} early in the process. Formally, the metric is computed individually per attribute as:

$$\begin{aligned} \mathrm{FD}\left(\boldsymbol{p}_{tar}^{(i)}, \boldsymbol{y}_{\tau_{bias}}^{(i)}\right) &= \left\|\boldsymbol{p}_{tar}^{(i)} - \boldsymbol{y}_t^{(i)}\right\|_2^2 \text{ with} \\ \boldsymbol{y}_t^{(i)} &= \frac{1}{N}\sum_{n=1}^{N} \eta_t^{(i)}\left(\boldsymbol{h}_t^n\right) \end{aligned} \tag{66}$$

and termed *estimated bias* for reference. Here, $\boldsymbol{y}_t^{(i)}$ is the mean prediction over the batch h-vectors $\boldsymbol{h}_t^{[1:N]}$ using h-space classifier $\eta_t^{(i)}$ for attribute $S^{(i)}$ and $\boldsymbol{p}_{tar}^{(i)}$ as the attribute's target distribution. Based on empirical analysis of the trained classifiers (see Section 6.1.2), a suitable timestep early in the process (in this case $\tau_{bias} = 31$) is selected to measure the bias of all attributes. The estimated bias subsequently serves as input to a function $f^{(i)} : I^{(i)} \to I_{content}$, where $I^{(i)} \subset \mathbb{R}$ is the range of expected bias values and $I_{content} \subset [1, T]$ the subset of timesteps considered to fall into the model's content generation stage (see Section 4.2). The intuition, as explained previously, is to map higher bias values to later timesteps, resulting in an earlier modification of the generative process. Consequently, $f^{(i)}$ is chosen to be a standard linear interpolation function. According to the derived bounds of the metric in Section 3.3, $I^{(gender)} = [0, 0.5]$, $I^{(race)} = [0, 0.75]$ and $I^{(age)} = [0, 1.125]$ are defined, and based on Section 4.2, $I_{content} = [30, 45]$ is selected. The linear interpolation functions are thus given as:

$$\begin{aligned} f^{(gender)}(b) &= \mathrm{round}\left(30 + \frac{(b-2)(45-30)}{5-2}\right) \\ &= \mathrm{round}\left(30 + 30b\right) \\ f^{(race)}(b) &= \mathrm{round}\left(30 + 20b\right) \\ f^{(age)}(b) &= \mathrm{round}\left(30 + \frac{40}{3}b\right) \end{aligned}$$

for an estimated bias value $b \in I_{bias}^{(i)}$ of attribute $S^{(i)}$ as measured by the h-space classifiers at timestep τ_{bias} and a classical integer rounding function round. This approach yields a timestep $\iota^{(i)} = f^{(i)}(b)$ at which to insert desired classes for each attribute. The next section will elucidate how this is performed. The full mechanism of detecting insertion timesteps $\iota^{(1:M)}$ is exemplified in Figure 7. Essentially, the FD metric measures a similarity between two discrete distributions which can be achieved as well using a range of other popular metrics (f.e. KL-divergence or Chi-square distance). We leave this as an opportunity for future research.

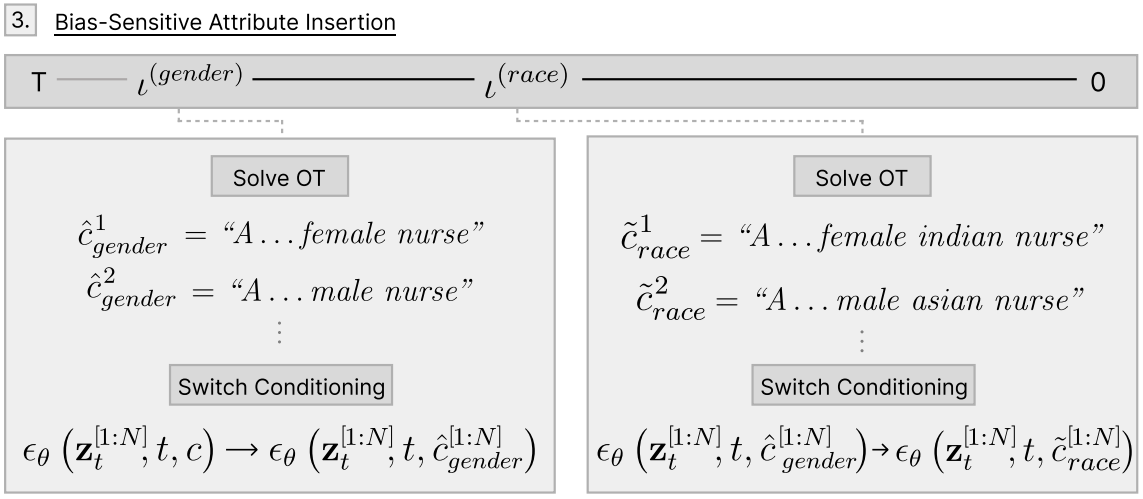


Figure 8: Schematic overview of the attribute insertion process in Debias Diffusion at the dynamically determined insertion timesteps $\iota^{(1:M)}$ (here $\iota^{(gender)}$ and $\iota^{(race)}$).

4.4.3 Bias-Sensitive Attribute Insertion

Upon completion of the generative process until τ_{bias} and computation of corresponding insertion timesteps $\iota^{(1:M)}$ for attributes $S^{(1:M)}$, the process continues from:

$$\hat{t} := \max\{\tau_{bias}, \iota_{max}\}$$

where $\iota_{max} := \max \iota^{(1:M)}$ is the earliest insertion timestep. Essentially, if $\hat{t} = \tau_{bias}$, generation continues until ι_{max} where the process is first modified. However, if $\hat{t} = \iota_{max} > \tau_{bias}$, the generative process has to restart at an earlier step. This can be done efficiently by re-using the cached latents $\boldsymbol{z}^{[1:N]}_{\iota_{max}}$ computed before to continue generation at ι_{max}. This prevents re-computing the latents for timesteps $[\iota_{max}, T]$.

The subsequent goal is to debias each sensitive attribute $S^{(1:M)}$ by inserting desired attribute classes into the prompt at timesteps $\iota^{(1:M)}$ dynamically per batch element. Refer to Figure 8 for an additional schematic illustration. In this example, $\iota_{max} = \iota^{(gender)} > \tau_{bias}$ meaning that the generative process resumed at $\iota^{(gender)}$ which was prior to τ_{bias} due to a pronounced gender bias in the batch under generation. Afterwards, it remains to determine a suitable class to insert for each batch element. For notational clarity, we explain the approach for a fixed attribute S. As noted by Parihar et al. (2024), this assignment can be optimized by incorporating each element's individual predictions. As an example, consider a binary class setup with two arbitrary batch samples classified as $\boldsymbol{p}^1 = [0.3, 0.7]$ and $\boldsymbol{p}^2 = [0.9, 0.1]$ and a balanced target distribution over both classes. With random label assignments, each batch element would have to switch its sensitive attribute's class 50% of the time. Instead, using the h-space classifiers class predictions, no element has to change if predicted correctly. In order to determine a suitable modifier for each batch element and attribute, consider a set of vectors

$$\boldsymbol{y}^{[1:N]} = \eta_t(\boldsymbol{h}_t^{[1:N]})$$

denoting the estimated class probability by the h-space classifier at timestep t. Additionally, the same number of one-hot vectors $\boldsymbol{o}^{[1:N]} \sim \boldsymbol{p}_{tar}$ are sampled according to the target

distribution. Inspired by X. Shen et al. (2024), the algorithm optimizes the class assignments per batch element by solving an *optimal transport (OT)* problem (Monge, 1781) between both sets of vectors. Formally, the optimal class indices are obtained by solving:

$$\boldsymbol{\sigma}^* \coloneqq \arg\min_{\boldsymbol{\sigma} \in \mathcal{P}_N} \sum_{n=1}^{N} \| \boldsymbol{y}^n - \boldsymbol{o}^{\sigma_n} \|_2$$

where $\mathcal{P}_N$ denotes all permutations of the integer range $|N| \coloneqq [1, \dots, N]$, $\boldsymbol{\sigma} = [\sigma_1, \cdots, \sigma_N]$ and $\sigma_n \in |N|$.

For multiple attributes $S^{(1:M)}$, the OT problem is solved independently for each attribute to obtain corresponding class assignment. Assume the generative process has reached (or been reset to) timestep $\iota_{max} = \iota^{(i)}$ and the attribute's desired classes $s^{[1:N]} \in S^{(i)}$ for each batch sample have been properly selected by solving the OT problem. Furthermore, consider the initial prompt c individually per batch element using $c^{[1:N]}$. The class s^n is inserted as an additional modifier into the prompt c^n prior to the human-related noun. This yields an individual *debiased prompt* $\hat{c}^n$ for each element (see Figure 8). To implement this practically, the syntactic representation from spaCy (see Section 4.4.1) is leveraged to insert the modifier prior to the noun in the prompt. After re-encoding the new prompts, generation continues with the new embeddings. This process can be performed independently for each attribute to obtain debiased prompts $\hat{\boldsymbol{c}}^{(1:M)}$ where we abbreviate the batch as $\hat{\boldsymbol{c}} \coloneqq \hat{c}^{[1:N]}$ for clarity. Essentially, every batch element employs an individual prompt, modified per attribute at the corresponding steps $\iota^{(1:M)}$. Consequently, sampling is performed according to a new distribution, formally given as:

$$\begin{aligned} &p_{\boldsymbol{\theta}}(\boldsymbol{z}_{0:T}, c, \hat{\boldsymbol{c}}^{(1:M)}) \coloneqq \\ &p(\boldsymbol{z}_T) \prod_{t=\iota^{(1)}+1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}, c \mid \boldsymbol{z}_t) \prod_{t=\iota^{(2)}+1}^{\iota^{(1)}} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}, \hat{\boldsymbol{c}}^{(1)} \mid \boldsymbol{z}_t) \cdot \ldots \cdot \prod_{t=0}^{\iota^{(M)}} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}, \hat{\boldsymbol{c}}^{(M)} \mid \boldsymbol{z}_t) \end{aligned} \tag{67}$$

assuming $\iota^{(M)} \leq \ldots \leq \iota^{(1)} = \iota_{max}$ without loss of generality. In essence, the vanilla generative process runs until τ_{bias}, suitable insertion timesteps $\iota^{(1:M)}$ are computed, and generation continues according to Equation (67). It is posited that, since each $\iota^{(i)}$ is chosen to be rather large, $p(\boldsymbol{z}_{t-1}, c \mid \boldsymbol{z}_t)$ as well as $p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}, \hat{c}^{(i)} \mid \boldsymbol{z}_t)$ for all i are very likely to overlap since they are both from the same family of "blurred", noisy distributions that are fully or largely supported over the ambient space (Y. Choi et al., 2024; Yin et al., 2024). Consequently, subsequent sampling operations should reside within the same data manifold as those prior to each $\iota^{(i)}$, preventing deteriorated out-of-distribution samples.

5 Evaluation

In this section, the presented method is thoroughly analyzed with regards to the proposed criteria (see Section 4.1). Simultaneously, multiple competing state-of-the-art debiasing algorithms for diffusion models are evaluated as well to provide proper context and baselines. Therefore, a comprehensive empirical evaluation is conducted with well over 1.5 million generated images. Sections 5.1, 5.2, and 5.3 detail the experimental setup, including models and metrics employed. In Section 5.4, each debiasing method[3] is evaluated on a wide range of different prompting scenarios to properly assess their general performance with respect to debiasing human subjects. Finally, in Section 5.5, a discussion of the results summarizes important insights, examining the implications of the findings on the developed method with respect to the proposed key factors (see Section 4.1).

5.1 Debiasing Methods

The following subsections provide a detailed examination of each analyzed method. To ensure fairness in comparison, all methods utilize the same fundamental pipeline and model parameters. The foundational model for each analysis is Stable Diffusion v1.5, identified as *runwayml/stable-diffusion-v1-5* on the HuggingFace Hub (runwayml, 2024).[4] This model is widely used in scientific research, offering reasonable comparison baselines to related works. Furthermore, it is one of the few large scale TTI diffusion models publicly available and employs a CC0 1.0 license that permits the usage of generated images for scientific purposes (Z. J. Wang et al., 2023; StabilityAI, 2024b). Practically, each debiasing method is based on and inherits from the *StableDiffusionPipeline* contained in HuggingFace's *diffusers* library (Platen et al., 2022).

5.1.1 Debias Diffusion

For a detailed explanation of the method, refer to Section 4.4. Building on insights from Section 6.1.2, we select $\tau_{bias} = 31$ as the timestep for measuring batch bias during generation. The h-space classifiers for all three attributes were trained on h-vectors obtained by generating 5000 images balanced across all sensitive groups. We employed the prompt template *"A photo of the face of a <occupation>, a person"* to generate diverse human face depictions, a setup similar to that used by X. Shen et al. (2024) for finetuning SD for fairness. We selected the first 500 occupations from their list of 1000 unique occupations, creating 10 images for each. Human faces were filtered, and sampling was repeated until the required number of data points was collected. The resulting h-vectors were labeled using image classifier predictions of the final images. For a more detailed explanation of the training procedure, see Section 4.3, and for more details on the face and classification models used, see Section 5.2. Using a single NVIDIA A40 GPU, the dataset creation took about 3 hours. Following this, each h-space attribute classifier was trained on the dataset for 100 epochs, taking approximately 37 minutes, with the same GPU setup.

[3] We denote the practical implementation of an algorithm as its method.

[4] Note that during the completion of this work, the model has been taken down from all available sources. A backup version has thus been provided in the work's source code.

5.1.2 Fair Diffusion

One of the pioneering approaches to debias the outputs of TTI diffusion models is *Fair Diffusion (FD)* (Friedrich et al., 2023). This semi-automatic method interferes with the model's generative process by leveraging its inherent semantic understanding for guidance purposes. The technique employed is known as Semantic Guidance (SEGA) (Brack et al., 2023), a general framework used to steer the generative process through textual descriptions of semantic concepts. Assume we aim to enforce or suppress the generation of a concept, such as $c_e =$ *"beard"*. To achieve this, the classifier-free guidance term (55) is extended to include a guidance term defined by this concept. Thus, the model's predictions are formulated as:

$$\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, c, c_e) = \underbrace{\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t) + s_g\left(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, c) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t)\right)}_{\text{classifier-free guidance}} + \underbrace{\gamma(\boldsymbol{z}_t, c_e)}_{\text{semantic guidance}} \tag{68}$$

where $s_g \in \mathbb{R}$ is the guidance scale. Note that in practice, $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ is also conditioned on t, which is omitted here for clarity. The semantic guidance term is implemented as proposed by Brack et al. (2023):

$$\gamma(\boldsymbol{z}_t, c_e) = \mu(\psi; s_e, \lambda)\psi(\boldsymbol{z}_t, c_e) \tag{69}$$

where ψ depends on the prescribed guidance direction for concept c_e:

$$\psi(\boldsymbol{z}_t, c_e) = \begin{cases} \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, c_e) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t), & \text{if positive guidance} \\ -(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, c_e) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t)), & \text{if negative guidance.} \end{cases} \tag{70}$$

To ensure a somewhat isolated effect, μ applies an element-wise scaling of magnitude s_e to the most relevant predicted values:

$$\mu(\psi; s_e, \lambda) = \begin{cases} s_e, & \text{where } |\psi| \geq \eta_\lambda\left(|\psi|\right) \\ 0, & \text{otherwise} \end{cases} \tag{71}$$

where $\eta_\lambda\left(|\psi|\right)$ defines the λ-th percentile of entries in ψ with the highest absolute values. Increasing s_e results in a stronger editing effect, while a higher λ leads to more isolated changes.

The theoretical motivation for this approach is akin to that of classifier-free guidance. Recall the score formulation from Section 2.4.2 and assume access to:

$$\begin{aligned} \boldsymbol{\epsilon}^*(\boldsymbol{z}_t, c_e) &= -\sqrt{1-\bar{\alpha}_t}\nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_t|c_e) \\ \boldsymbol{\epsilon}^*(\boldsymbol{z}_t) &= -\sqrt{1-\bar{\alpha}_t}\nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_t) \end{aligned} \tag{72}$$

which are exact scores of the true conditional $p(\boldsymbol{z}_t|c_e)$ and marginal $p(\boldsymbol{z}_t)$ distributions, respectively. These distributions are defined as expectations over the forward process:

$$\begin{aligned} p(\boldsymbol{z}_t|c_e) &= \mathbb{E}_{\boldsymbol{z}_0 \sim q(\boldsymbol{z}_0|c_e)}\left[q(\boldsymbol{z}_t|\boldsymbol{z}_0)\right] \\ p(\boldsymbol{z}_t) &= \mathbb{E}_{\boldsymbol{z}_0 \sim q(\boldsymbol{z}_0)}\left[q(\boldsymbol{z}_t|\boldsymbol{z}_0)\right]. \end{aligned} \tag{73}$$

Here, $q(\boldsymbol{z}_0)$ and $q(\boldsymbol{z}_0|c_e)$ are distributions obtained by perceptual compression of real data points $\boldsymbol{x} \sim p(\boldsymbol{x})$ and $\boldsymbol{x} \sim p(\boldsymbol{x}|c_e)$ in the LDM framework (see Section 2.4.4). Now, consider the implicit classifier:

$$p(c_e|\boldsymbol{z}_t) = \frac{p(\boldsymbol{z}_t|c_e)}{p(\boldsymbol{z}_t)} \tag{74}$$

for concept c_e, obtained by inverting the generative model using Bayes' rule. The (negative) score of this implicit classifier is:

$$\begin{aligned} -\nabla_{\boldsymbol{z}_t} \log p(c_e|\boldsymbol{z}_t) &\overset{(74)}{=} -\nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_t|c_e) + \nabla_{\boldsymbol{z}_t} \log p(\boldsymbol{z}_t) \\ &\overset{(72)}{=} \frac{1}{\sqrt{1-\bar{\alpha}_t}} \left(\boldsymbol{\epsilon}^*(\boldsymbol{z}_t, c_e) - \boldsymbol{\epsilon}^*(\boldsymbol{z}_t)\right). \end{aligned} \tag{75}$$

Essentially, this score vector points in the direction of the latent space with a higher likelihood of containing concept c_e according to the implicit classifier. Replacing the noise prediction in Equation (68) with the exact ones, and denoting the classifier-free guidance term as $\bar{\epsilon}^*(\boldsymbol{z}_t, c)$, we have:

$$\begin{aligned} \boldsymbol{\epsilon}^*(\boldsymbol{z}_t, c, c_e) &= \bar{\boldsymbol{\epsilon}}^*(\boldsymbol{z}_t, c) - s_e\sqrt{1-\bar{\alpha}_t}\nabla_{\boldsymbol{z}_t} \log p(c_e|\boldsymbol{z}_t) \\ &\overset{(75)}{=} \bar{\boldsymbol{\epsilon}}^*(\boldsymbol{z}_t, c) + s_e \left(\boldsymbol{\epsilon}^*(\boldsymbol{z}_t, c_e) - \boldsymbol{\epsilon}^*(\boldsymbol{z}_t)\right) \end{aligned} \tag{76}$$

Note how Equation (76) resembles the classifier-free guidance term in Equation (68) with the "base" estimate $\bar{\boldsymbol{\epsilon}}^*$ already conditioned on c . However, this is merely a theoretical inspiration since $\boldsymbol{\epsilon}^*$ and $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$ are fundamentally different. The former is the (scaled) gradient of a hypothetical, optimal classifier, contrasting with $\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}$, which is the output of an unconstrained neural network. Thus, there are no theoretical guarantees for SEGA's effectiveness. Nevertheless, empirical results from Brack et al. (2023) and Section 5.4 demonstrate this approach's efficacy.

Fair Diffusion uses the SEGA technique for debiasing by employing classes of each sensitive attribute $S^{(i)}$ as concepts c_e. For instance, in the original work, the authors used concepts $c_1 =$ *"female person"* and $c_2 =$ *"male person"* to debias gender depiction by randomly enforcing one and suppressing the other with equal probability. This procedure is similarly applied to race and age (if they are being debiased). For non-binary attributes such as race, one class is enforced and all others are suppressed. This requires an additional forward pass through the UNet for each applied class to compute the score predictions. As with vanilla classifier-free guidance, this can be efficiently achieved (with sufficient VRAM) by concatenating all latents and performing a single batch pass through the UNet. However, this technique still results in significantly decreased generation speed and increased memory consumption, as demonstrated in Section 5.4. Furthermore, the authors found it beneficial to allow for a warm-up parameter $\delta \in [1, T]$, after which guidance is first applied. Additionally, a momentum term can be accumulated over past guidance terms (even during warm-up) to allow for less volatile changes. For more details on these parameters, refer to Brack et al. (2023).

On a practical level, Fair Diffusion is implemented by modifying the *SemanticStableDiffusionPipeline* contained in the diffusers library. It represents a StableDiffusionPipeline with support for SEGA. Fair Diffusion, however, is semi-automatic, requiring a lookup table of words or subjects signaling a need for debiasing, such as a list of occupations. Moreover, the method necessitates defining a range of parameters $(s_e, \lambda, \delta, \ldots)$ related to each attribute, with no guidance on how to approach this. Thus, we adopted the values provided by Friedrich et al. (2023) for their gender debiasing examples. For race and age, parameters were chosen such that generated results for highly biased and balanced domains appeared satisfactory.

5.1.3 Attribute Switching

In their work, Friedrich et al. (2023) emphasize the motivation to avoid text-based inference in prompts, a concern addressed by the SEGA technique. This method allows for textual guidance without relying on additional language understanding capabilities. Building on early approaches in this domain (Bansal et al., 2022), Friedrich et al. (2023) questioned the effectiveness of using textual interventions to steer the generative process. However, as demonstrated in Section 5.4, Debias Diffusion can empirically dispel these concerns. In this regard, inspiration was drawn from the experiments conducted by Y. Choi et al. (2024). The authors present a theoretical and empirical analysis on debiasing diffusion model outputs concerning a binary sensitive attribute by switching class conditioning during generation. Focusing on their approach to TTI diffusion models, we translate their formulations from score-prediction to noise-prediction models for consistency with the rest of this work. We assume that $-\nabla_{\boldsymbol{z}_t} \log p_{\boldsymbol{\theta}}(\boldsymbol{z}_t) = \epsilon_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t)$, omitting the factor $1/\sqrt{1-\bar{\alpha}_t}$ in Equation (47), inferred from the source code provided by Y. Choi et al. (2024). As further confirmation, when this factor is included, the results exhibit significantly different qualities compared to those obtained by the original authors. However, when the factor is excluded, the results align closely with those reported in their study (see Figure 9). Following Y. Choi et al. (2024), the categorical class conditioning is replaced with text conditioning using two prompts c_1 and c_2, defined as *"A color photo of the face of a <s_1/s_2>"* for either class $s_1, s_2 \in S$ of the given binary sensitive attribute. It is advised to use classes {*"man"*, *"woman"*} for gender and {*"young person"*, *"old person"*} for age (Y. Choi et al., 2024). This approach is extended to non-binary attributes such as race later in this section.

As explained in Section 4.2, Y. Choi et al. (2024) hypothesize that it is possible to retain high-level features related to c_1 while simultaneously generating an image from $p(\boldsymbol{x}|c_2)$ by switching the text conditioning at a suitable timestep $\tau \in [1, T]$ during generation. Without loss of generality, we assume a switch in the direction from c_1 to c_2 throughout this section. The authors present a theoretically grounded approach to yield an optimal switching step τ. Consider the following difference of noise predictions at timestep t:

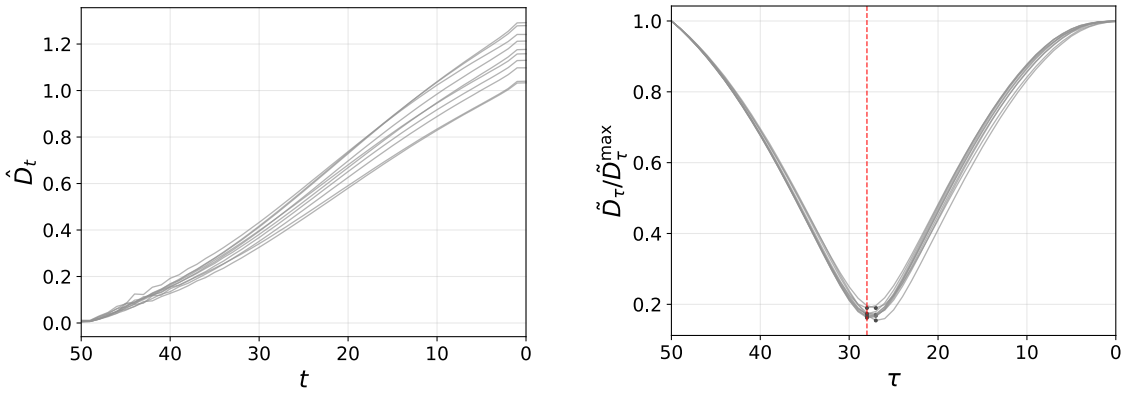
$$D_t \coloneqq \beta_t \underbrace{(\hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c_1) - \hat{\boldsymbol{\epsilon}}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, t, c_2))}_{\hat{D}_t} \tag{77}$$

where β_t is the variance of the forward process. The objective is to find τ that satisfies:

$$\tau = \arg\min_{\tau} \underbrace{\left\| \sum_{t \leq \tau} D_t - \sum_{t \geq \tau} D_t \right\|}_{\tilde{D}_\tau} \tag{78}$$

Y. Choi et al. (2024) provide a theoretical proof ensuring the independence of the resulting distribution from attribute S for infinitesimally small timesteps when switching at τ. Subsequently, fair sampling is performed according to:

$$p_{\boldsymbol{\theta}}(\boldsymbol{z}_{0:T}, c_1, c_2) = p(\boldsymbol{z}_T) \prod_{t=\tau+1}^{T} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t, c_1) \prod_{t=0}^{\tau} p_{\boldsymbol{\theta}}(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t, c_2). \tag{79}$$



(a) Depiction of the difference in noise predictions $\hat{D}_t$ as defined in Equation (77).

(b) Depiction of the normalized differences $\tilde{D}_\tau$ as defind in Equation (78).

Figure 9: Each line plot was computed as an average over a batch of 5/10 samples. For each batch element, two sets of predictions over all timesteps were computed using the same initial latents but different text conditionings.

The condition in Equation (78) might not intuitively clarify the optimal switching point. Consider the differences in score predictions from Equation (77), denoted as $\hat{D}_t$. Both latent trajectories start at the same point $\boldsymbol{z}_T$. The intuition is that $\hat{D}_t$ gradually increases since both predictions are conditioned on different prompts $c_1 \neq c_2$, causing the latents to pursue diverging trajectories. This behavior is reflected in Figure 9a, where we plot the mean $\hat{D}_t$ over a batch of 5 generations for 10 different runs. The hypothesized (almost) monotonically increasing difference in the predictions is evident. For a more detailed explanation of this phenomenon, refer to Section 4.2. On the other hand, the variances $\beta_{[1:T]}$ are monotonically decreasing for $t \to 1$. Therefore, D_t is a concave function of t, and subsequently, $\tilde{D}_t$ from Equation (78) is convex, as illustrated in Figure 9b, where $\tilde{D}_{max} \coloneqq \max \tilde{D}_{[1:T]}$. Regarding the normalization in Figure 9b, note that dividing a convex function by a positive constant always yields another convex function. As the variances decrease, the latents settle on certain features related to c_1. $\hat{D}_t$ provides a way to measure how similar both sets of features are. As such, the optimal switch step τ provides the best theoretical timepoint to nudge the trajectory towards a path that ends in a sample from $p(\boldsymbol{x}|c_2)$ while rendering rich features related to c_1 (Y. Choi et al., 2024).

Y. Choi et al. (2024) focus their work on an isolated setting for class-conditional diffusion models. Furthermore, no instructions are provided on how to extend their algorithm to non-binary as well as multiple attributes or how to automate it for TTI models. To enable a fair comparison to the other methods, the employed method inherits the syntactic capabilities of Debias Diffusion and builds on the same pipeline. As a consequence, classes for sensitive attributes are automatically inserted into the prompt using spaCy and NLTK (see Section 4.4.1). Suitable insertion timesteps $\tau^{(1:M)}$ are obtained according to Equation (79) for each sensitive attribute $S^{(i)}$. The authors advise using 4 batches of 5 elements each for experiments with SD, which we found to be suitable regarding the small amount of variance between batch solutions (see Figure 9b). For non-binary attributes such as race, we create conditioning prompts c_1 and c_2 by randomly sampling two classes $s_1, s_2 \in S^{(i)}$

for each batch element and sensitive attribute. Therefore, we also increase the number of batch elements used to compute τ in Equation (79) to 10. This approach yields results with similar qualities and variances as for the binary case. The resulting switching points are given as $\tau_{gender} = 28$, $\tau_{age} = 29$, and $\tau_{race} = 28$. Note that two UNet passes are necessary per timestep and batch element to obtain the latents' trajectories for either prompt. This procedure takes around one minute per attribute on a single A40 GPU.

Sampling for multiple attributes is performed according to Equation (67) using the fixed pre-computed timesteps $\tau^{(1:M)}$. Similar to Fair Diffusion, the class to insert is randomly sampled according to the target distribution. Since no classifiers are involved, no attribute state can be predicted and therefore a switch from an empty class $s_1 = \emptyset$ to some class $s_2 \in S^{(i)}$ is assumed. The choice of class is weighted by the target probability for that attribute. This method, denoted as *Attribute Switching (AS)*, basically resembles Debias Diffusion without h-space classifiers (see Section 4.3) and bias-dependent batch distribution guidance (see Sections 4.4.2 and 4.4.3) instead using the theoretically optimal switching timestep as advised by Y. Choi et al. (2024). Therefore, it enables a rather isolated analysis of the proposed method used to dynamically compute bias-guided insertion steps during generation.

5.1.4 Finetuning Diffusion Models for Fairness

The final method employed for comparison is the approach developed by X. Shen et al. (2024), which focuses on the training rather than the deployment stage, as the previously discussed methods do. This technique involves finetuning existing model parameters with a loss function specifically designed to enhance the fairness of human face depictions. We refer to this method as *FDM (**F**inetuning **D**iffusion **M**odels for Fairness)*. For notational simplicity, we assume a fixed sensitive attribute $S = [s_1, \dots, s_K]$ and note that the algorithm can be applied analogously to other attributes. To elucidate the approach, consider a discrete target distribution $\boldsymbol{p}_{tar}$ over the attribute's classes and a batch of generated images $\boldsymbol{x}^{[1:N]}$. The objective is to align the distribution of sensitive classes $s_{[1:K]}$ with $\boldsymbol{p}_{tar}$ while maintaining the original image semantics. To achieve the prescribed goal, X. Shen et al. (2024) propose a loss function comprising three components, detailed as follows.

The first component involves sampling a one-hot vector $\boldsymbol{u}^{[1:N]} \subset \{0,1\}^K$ for each batch element such that $\boldsymbol{u}^{[1:N]} \sim \boldsymbol{p}_{tar}$. Likewise, an image classifier $\mathcal{C} : \mathcal{X} \mapsto [0,1]^K$ is employed to estimate the model's generated (discrete) distribution $\boldsymbol{p}_{\boldsymbol{\theta}}$. Class probabilities for each image are formally obtained as $\boldsymbol{y}^n = [y_1^n, \dots, y_K^n] = \mathcal{C}(\boldsymbol{x}^n)$ (see Section 5.2 for the classifiers used). Optimal assignments $\boldsymbol{\sigma}^* = [\sigma_1^*, \dots, \sigma_N^*]$ between predicted and target attribute states, $\boldsymbol{y}^{[1:N]}$ and $\boldsymbol{u}^{[1:N]}$, are determined by solving an optimal transport problem, similar to Section 4.4.3. To avoid a collapsed target distribution by employing single one-hot targets, several sets of one-hot vectors are sampled per batch. The resulting optimal assignments are used to compute a Monte-Carlo estimate of the target probabilites. This is formally obtained as:

$$\boldsymbol{q}^n = \mathbb{E}_{\boldsymbol{u}^n \sim \boldsymbol{p}_{tar}} \left[\boldsymbol{u}^{\sigma_n^*} \right] \in [0,1]^K \quad \forall n \in |N|.$$

$\boldsymbol{q}^{[1:N]}$ represent the probabilities for each batch element to belong to one of the K classes, assuming the batch of generated images had indeed followed $\boldsymbol{p}_{tar}$. Essentially, $\boldsymbol{q}^n$ is a

weighted average over optimal target one-hot vectors for batch element n. Consequently, the target classes $q^n = \arg\max(\boldsymbol{q}^n)$, encoded as one-hot vectors $\tilde{\boldsymbol{q}}^n$, and attribute predictions $\boldsymbol{y}^n$ are used to form the *distributional alignment loss* as follows:

$$\mathcal{L}_{align} \coloneqq \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}(c^n \geq C) \mathcal{L}_{CE}\left(\boldsymbol{y}^n, \tilde{\boldsymbol{q}}^n\right)$$

Here, $\mathbb{1}$ denotes the indicator function, which excludes batch samples from influencing the loss where the probability $c^n = \max(\boldsymbol{q}^n)$ is below some pre-defined threshold $C \in [0, 1)$. $\mathcal{L}_{CE}$ is a standard cross-entropy loss defined as:

$$\mathcal{L}_{CE}(y, q) = -(y \log(q) + (1 - y) \log(1 - q))$$

The second component focuses on preserving original image semantics. Assume access to a copied, frozen version of the original diffusion model. Each time the finetuned model generates a batch of images $\boldsymbol{x}^{[1:N]}$, the frozen model generates a batch $\boldsymbol{x}_{orig}^{[1:N]}$ using the same prompts and seeds. Afterwards, the image semantics preservation loss is computed as:

$$\begin{aligned}\mathcal{L}_{img} \coloneqq \frac{1}{N} \sum_{n=1}^{N} \Bigg[&\left(1 - \cos\left(\text{CLIP}\left(\boldsymbol{x}^n\right), \text{CLIP}\left(\boldsymbol{x}_{orig}^n\right)\right)\right) \\ &+ \left(1 - \cos\left(\text{DINO}\left(\boldsymbol{x}^n\right), \text{DINO}\left(\boldsymbol{x}_{orig}^n\right)\right)\right) \Bigg]\end{aligned}$$

where CLIP (Radford et al., 2021) and DINO (Caron et al., 2021) are multimodal models used to measure image similarity (see Section 5.3). The cosine similarity $\cos(\cdot)$ quantifies embedding similarities, with $-1/1$ indicating perfect dis-/similarity.

Lastly, X. Shen et al. (2024) introduce a *face realism preserving loss* specific to human faces:

$$\mathcal{L}_{face} \coloneqq \frac{1}{N} \sum_{n=1}^{N} \left(1 - \min_{F \in D_F} \cos\left(\text{emb}\left(d_{face}\left(\boldsymbol{x}^n\right)\right), \text{emb}(F)\right)\right)$$

where $\text{emb}(\cdot)$ and $d_{face}(\cdot)$ are face embedding and detection models, respectively. D_F is an external set of high-quality images depicting diverse human faces. The loss is minimized if detected faces in generated images closely resemble real faces (in the embedding space). The final loss is defined as:

$$\mathcal{L} \coloneqq \mathcal{L}_{align} + \lambda_{img} \mathcal{L}_{img} + \lambda_{face} \mathcal{L}_{face}$$

with hyperparameters λ_{img} and λ_{face}. The complete finetuning algorithm includes additional components such as pixel-based gradient stops and dynamic hyperparameters. For a full description, refer to X. Shen et al. (2024).

The prescribed loss propagates gradients to either the model's UNet, text encoder, or both. The best trade-off between debiasing efficacy and image quality was achieved by tuning only the text encoder's parameters, highlighting the potential of leveraging language understanding capabilities for debiasing. Finetuning SD's text encoder took approximately 48 hours on eight NVIDIA A100 GPUs (X. Shen et al., 2024). For the face detection and

image classification models used, refer to Section 5.2. The dataset used for diverse, labeled, high-quality human face images was the FairFace dataset (Kärkkäinen and Joo, 2021), which is also employed in the proposed evaluation (see Section 5.3.2). To holistically debias the model, the authors employ the prompt template *"A photo of the face of a <occupation>, a person"* with a list of 1000 unique occupations for finetuning. Using the presented method for evaluation is as simple as loading the StableDiffusionPipeline and replacing the text encoder's weights. No other modifications are necessary concerning the generative process. Since the evaluation methodology was inspired by X. Shen et al. (2024), finetuned parameters for most debiasing experiments are readily available.

5.2 Face Detection and Image Classification Models

This section outlines the face detection and image classification models employed in this work, particularly for training the h-space classifiers (see Secion 4.3) and conducting evaluations (see Sections 5.4 and 6). These models are crucial for accurately classifying attributes such as race, age, and gender, which is integral to a proper evaluation.

The race and age classifiers employed in this work utilize models trained by X. Shen et al. (2024) on the FairFace dataset (Kärkkäinen and Joo, 2021). The original FairFace dataset categorizes race into seven classes: White, Black, Southeast Asian, East Asian, Middle Eastern, Indian, and Latino Hispanic. For the purposes of our evaluation, these classes are grouped into four broader categories: White (grouping White, Latino Hispanic and Middle Eastern), Black, Asian (comprising Southeast Asian and East Asian), and Indian. Similarly, the age attribute in the FairFace dataset is originally divided into nine classes: 0-2, 3-9, 10-19, 20-29, 30-39, 40-49, 50-59, 60-69, and 70+. Akin to race, X. Shen et al. (2024) consolidate these into two primary categories: Young (0-39) and Old (40+). For gender classification, the original model from X. Shen et al. (2024) was trained on the CelebA dataset (Z. Liu et al., 2015). However, this model demonstrated suboptimal performance on diverse datasets. Consequently, we employ the gender classifier developed by the FairFace dataset authors, which has shown greater accuracy and reliability in diverse contexts. The classifiers utilize the MobileNetV3 architecture (Howard et al., 2019), which offers a balance of computational efficiency and high performance.

Face detection is performed using the *InsightFace* library (An et al., 2022), specifically the *FaceAnalysis* module configured with the *buffalo_l* model. This model is optimized for CUDA execution, ensuring efficient processing on available NVIDIA GPUs. The face detection pipeline includes detecting and aligning faces within an image, expanding bounding boxes and aligning facial landmarks to standardize input for classification.

These models and their configurations are consistently applied across all experiments and evaluations in this work. Well performing models are essential for maintaining consistency and accuracy in automated attribute evaluations as well as the training of h-space classifiers. Thus, a lot of alternatives were tested and compared against each other. An important criteria for both classification and detection models was to perform similarly well on combinations of minority classes for all attributes. As such, regarding the domain of human faces, the FairFace dataset and subsequently models trained on it seem to perform better than comparable alternatives at the time of writing.

5.3 Metrics

To effectively quantify the proposed evaluation criteria for debiasing algorithms (see Section 4.1), it is crucial to employ suitable metrics. We utilize three broad criteria, each with at least one corresponding evaluation metric.

The primary objective of a debiasing method is to achieve fairness in accordance with stakeholders' target distributions. To quantify this, we employ the FD metric as defined in Equation (**??**). For a comprehensive description of this metric, refer to Section 3.3. It is important to note that the FD metric primarily captures the average fairness achieved across the set of analyzed images. However, this approach does not fully address considerations related to group fairness. To mitigate this limitation and provide a more nuanced comparison between methods, we additionally report the standard deviation of the metric with respect to each prompt. For instance, in Section 5.4.1, a prompt and therefore a group would comprise all images generated for a specific occupation.

5.3.1 Semantics Preservation

The second criterion under consideration is the preservation of the original model's semantics. While changing the model's inherent distribution necessitates some modification of the generated output, an effective debiasing algorithm should act as isolated as possible. Therefore, outputs of debiasing methods are evaluated based on their adherence to the input prompt and their similarity to the original generations.

Prompt adherence is quantified using the CLIP-Score (Hessel et al., 2021), which measures the cosine distance between the prompt and image embeddings of the CLIP model (Radford et al., 2021). Formally, for a generated image $\boldsymbol{x}$ and corresponding prompt c, it is defined as:

$$\text{CLIP-T}(\boldsymbol{x}, c) \coloneqq \omega \max\left(\cos\left(C_I(\boldsymbol{x}), C_T(c)\right), 0\right) \tag{80}$$

where C_T and C_I are the model's text and image encoders, respectively. Hessel et al. (2021) recommend using $\omega = 2.5$ to scale results to the range $[0, 1]$, as original cosine similarities typically fall within $[0, 0.4]$. In each experiment, the final value is reported as the average CLIP-Score across all generated samples. This metric has been empirically shown to align closely with human judgment, particularly for prompts beginning with phrases like *"A photo of..."* as used in Section 5.4.1 (Hessel et al., 2021).

Another aspect of semantics preservation is the perceptual similarity between original and modified image outputs. As demonstrated by related works (X. Shen et al., 2024), this can be effectively measured by comparing the CLIP image embeddings of modified and original image pairs $(\boldsymbol{x}, \tilde{\boldsymbol{x}})$. Similar to Equation (80), it is computed as:

$$\text{CLIP-I}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) \coloneqq \max\left(\cos\left(C_I(\boldsymbol{x}), C_I(\tilde{\boldsymbol{x}})\right), 0\right)$$

where the average is reported over all pairs.

However, similarity measurements using deep neural networks' feature representations are not limited to multimodal models like CLIP. Zhang et al. (2018) empirically demonstrated that this is a common emergent property shared across other deep network architectures

as well. At the time of their study, internal activations of state-of-the-art deep convolutional image classification networks, such as *VGG-Net* (Simonyan and Zisserman, 2015), were used for this purpose. Zhang et al. (2018) observed empirically that similarity measurements between these activations correspond significantly better to human similarity judgments than previous metrics. To explain their proposed metric, consider two images $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ and an encoder network $\mathcal{E}$. The authors extract feature outputs from the first L layers of $\mathcal{E}$ and normalize them over the channel dimensions. This yields vectors $\boldsymbol{y}^l, \hat{\boldsymbol{y}}^l \in \mathbb{R}^{H_l \times W_l \times C_l}$ for all L layers, where H_l, W_l are the spatial and C_l the channel dimensions of each layer. The actual metric, termed *Learned Perceptual Image Patch Similarity (LPIPS)*, is computed as:

$$\text{LPIPS}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \sum_{l=1}^{L} \frac{1}{H_l W_l} \sum_{h=1}^{H_l} \sum_{w=1}^{W_l} \left\| \omega_l \left(\boldsymbol{y}^l_{hw} - \hat{\boldsymbol{y}}^l_{hw} \right) \right\|_2^2$$

with an optional channel-wise scaling of network activations by $\omega_{[1:L]}$ (Zhang et al., 2018). Note that setting $\omega_1 = \ldots = \omega_L = 1$ is equivalent to computing the cosine-distance between the activations, which is the approach used throughout this work.

Both CLIP-I and LPIPS have their limitations but remain widely used metrics. For a proper interpretation of the results, it is essential to consider the biases inherent to CLIP (see Section 3) as well as the sensitivity of LPIPS to image perturbations (Ghazanfari et al., 2024). By employing both metrics simultaneously, these factors can be better accounted for through cross-comparison of results. In practice, we use the LPIPS implementation from the *PIQ* library (Kastryulin et al., 2022) and their pre-trained VGG16 model for feature extraction. For CLIP, the largest currently available model is used, identified as *openai/clip-vit-large-patch14* on the HuggingFace Hub (OpenAI, 2024).

5.3.2 Image Quality

The final evaluation criterion focuses on image quality. Objectively and automatically evaluating the quality of generative models' outputs is inherently challenging (Stein et al., 2023). Common approaches address this by framing it as a distributional alignment problem (Salimans et al., 2016; Heusel et al., 2017; Chong and Forsyth, 2020; Baevski et al., 2022). Two key aspects are considered: **fidelity**, which assesses how closely the generated data mirrors the original training set, and **diversity**, which evaluates how well the generated samples cover the full range of variations within the training data. Most current metrics aim to measure both simultaneously, often without allowing for a flexible trade-off, likely to facilitate method rankings (Stein et al., 2023). These concepts are illustrated in Figure 10.

Similar to image semantics, most contemporary image quality metrics evaluate images in a compressed representation space (Salimans et al., 2016; Heusel et al., 2017; Nash et al., 2021; Jiralerspong et al., 2023). For instance, the widely used *Fréchet Inception Distance (FID)* (Heusel et al., 2017) utilizes embeddings from the *InceptionV3* network (Szegedy et al., 2016), a classification model trained on the *ImageNet-1K* dataset (Deng et al., 2009). Formally, consider a set of generated images $\tilde{\boldsymbol{x}}^{[1:N]}$ and training images $\boldsymbol{x}^{[1:N]}$. Let $(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}})$ represent the means and variances of the sets' encoded representations

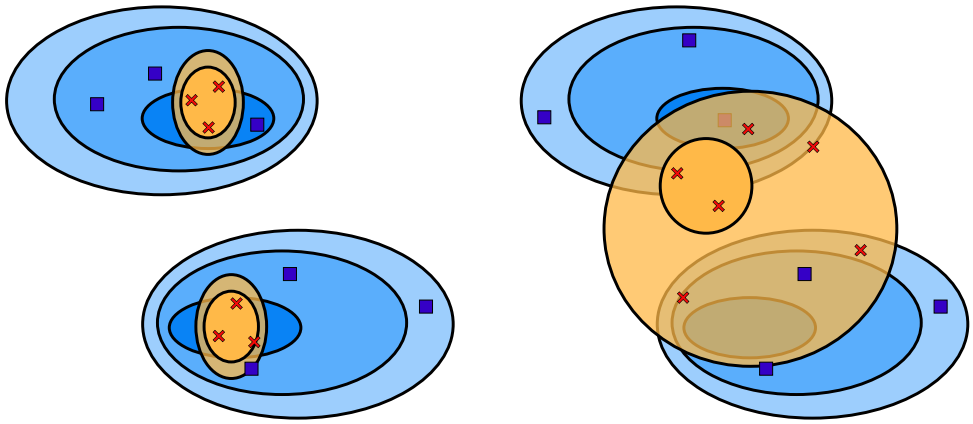


Figure 10: Illustration of the concepts of fidelity (left) and diversity (right). The learned distributions (orange circles) and corresponding generated samples (red crosses) exhibit different characteristics relative to the true distributions (blue) of training/real samples (blue squares). Higher fidelity is characterized by a learned distribution closer to prominent modes in training data, while higher diversity provides better coverage of the training distribution. The illustration is recreated from Stein et al. (2023).

(using InceptionV3). The metric is computed as:

$$\text{FID}(\boldsymbol{\mu}, \boldsymbol{\Sigma}, \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}) = \|\boldsymbol{\mu} - \tilde{\boldsymbol{\mu}}\|_2^2 + Tr\left(\boldsymbol{\Sigma} + \tilde{\boldsymbol{\Sigma}} - 2\left(\boldsymbol{\Sigma}\tilde{\boldsymbol{\Sigma}}\right)^{\frac{1}{2}}\right)$$

where Tr denotes the matrix-trace operator (Stein et al., 2023). This metric resembles the Wasserstein-2 distance between multivariate normal distributions. Thus, if both distributions are holistically characterized by their first two moments, as is the case for Gaussians, FID constitutes a proper metric between both distributions. Despite its widespread use, FID has been reported to have several shortcomings, such as susceptibility to distortions (Kynkäänniemi et al., 2023), bias towards texture and shape (Chong and Forsyth, 2020; Y. Choi et al., 2024), and other issues (Morozov et al., 2021; Parmar et al., 2022; Hong et al., 2022). Notably, Chong and Forsyth (2020) demonstrated that the metric is biased and inaccurate for finite sample sizes. Furthermore, this sensitivity cannot be mitigated by fixing the sample size for all models since the bias also depends on the generator itself. Therefore, we use a modified FID metric as proposed by the authors. Chong and Forsyth (2020) suggest using multiple subsets of the employed datasets at regular intervals to fit a linear trend to the outputs and extrapolate the results for infinitely sized datasets. This method is denoted as FID_∞. Additionally, Stein et al. (2023) reported that Fréchet Distance measurements using InceptionV3 embeddings do not correlate well with human judgment and unfairly penalize diffusion models compared to other generative architectures. They propose using the DINO model (Caron et al., 2021) for computing representations as it empirically aligns better with human perception. Consequently, we report the FID_∞ metric by using the DINO model instead of InceptionV3 and rename it FD_∞ accordingly.

Lastly, while it may be natural to assume the generated and target distributions are Gaussians, there is no guarantee for this. Furthermore, the metric employs a feature encoder (DINO) which constitutes a biased estimator. We employ another image quality

metric that is independent of both. The *Kernel Distance (KD)* (Baevski et al., 2022) between generated and training images is defined as:

$$\mathrm{KD}\left(\tilde{\boldsymbol{x}}^{[1:N]}, \boldsymbol{x}^{[1:M]}\right) = \frac{1}{N(N-1)} \sum_{i \neq j}^{N} k(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j) + \frac{1}{M(M+1)} \sum_{i \neq j}^{M} k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) - \frac{2}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} k\left(\tilde{\boldsymbol{x}}_i, \boldsymbol{x}_j\right)$$

where k is generally a positive definite kernel, chosen to be a standard 3rd degree polynomial in this work. The metric provides a suitable alternative to FD_∞ but is not without its flaws either. For example, it has been criticised for having larger variances than competing alternatives (Ravuri and Vinyals, 2019).

Typically, these metrics are used in a controlled setting with an (un-)conditional diffusion model trained on readily available datasets. Applying them to the TTI model under investigation, Stable Diffusion, presents two major challenges: Firstly, its training dataset, LAION-400m (Birhane et al., 2021), is huge in size and not feasible to download and work with for this thesis. Secondly, both distributions should be somewhat similar, which is difficult to ensure since the evaluation aims to cover a broad spectrum of images, largely dominated by diverse depictions of human faces. The goal of the employed metric is to quantify image quality by comparing distributions of the images or their embedded features. As such, other structural aspects of the generated images may deteriorate the results or prevent an isolated assessment of quality. For example, consider a reference dataset that consists mostly of Caucasian-appearing faces. Assuming the goal is to generate balanced race depictions, methods producing more balanced results will likely be penalized by the metrics simply because there are fewer Caucasian-related features present overall[5] (Y. Choi et al., 2024). Following Parihar et al. (2024), we use a dataset of real images depicting human faces that aligns with the target distributions over all sensitive attributes. Similar to the authors, we found the FairFace dataset to be the best alternative in this regard. Furthermore, to compute FD_∞ and KD scores for the original model, we sampled data according to SD's generated distribution over sensitive attributes. This led to notably smaller datasets as the model, for example, generated more male, Caucasian-appearing people than contained inside FairFace. Thus, we marked these values in the corresponding tables and advise comparing them with caution, despite the additional robustness improvements of the employed metrics. Computing quality metrics for the original model's generations using a balanced reference dataset produced overly bad results, likely due to the distributional misalignment. Therefore, employed quality metrics also provide a notion of how well each model mimics the targets distribution (in the embedding space), simply due to their nature as distance measures between distributions (Y. Choi et al., 2024). Lastly, we employ the DINO-V2 model, identified as *facebook/dinov2-large* on the HuggingFace Hub, to compute FD_∞ scores (Facebook, 2024). Both metrics are calculated using the comprehensive open-source library provided by Stein et al. (2023).

[5] Implicitly assuming that Caucasian is the majority class in this example.

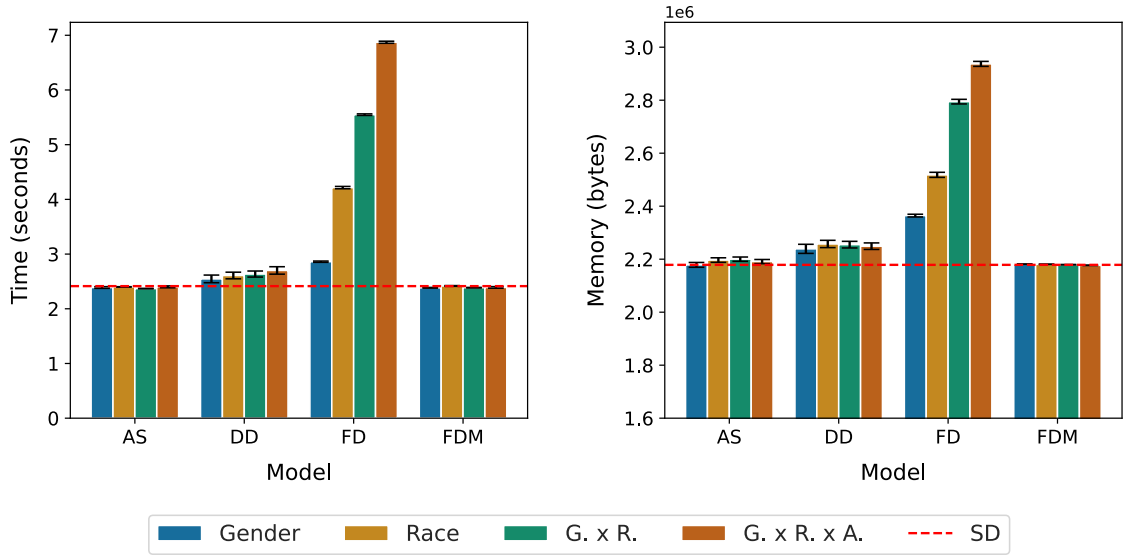


Figure 11: The plot on the left displays the average time per image and the plot on the right the average VRAM consumption per method in any given debiasing setup. The red dotted reference line indicates the values of the unmodified base model.

5.4 Results

This section presents a elaborate analysis of each method's performance across various usage scenarios, sensitive attributes, and target distributions. We provide both quantitative and qualitative results, followed by discussions on emergent properties to elucidate potential underlying mechanisms of the evaluated methods.

Throughout the experiments, the analysis is focused on the following sensitive attributes and their combinations: gender, race, gender and race (GxR), and gender, race, and age (GxRxA). For brevity, we refer to these as *debiasing setups*. Attributes included in a given debiasing setup are termed *debiased attributes*. It is important to note that, in line with current research practices (Friedrich et al., 2023; Y. Choi et al., 2024; X. Shen et al., 2024; J. Li et al., 2024), we employ binary gender classifications, acknowledging the inherent limitations of this approach. Following common practices in fairness literature (X. Shen et al., 2024; Parihar et al., 2024), we categorize race into four classes: White, Black, Asian, and Indian, while recognizing that these categories may not fully encompass all identities or ethnicities. The age attribute is classified into Young and Old, based on the classifier described in Section 5.2. For gender and race, we aim for equal representation among all classes, aligning with related literature (X. Shen et al., 2024; Parihar et al., 2024; Cortes and Vapnik, 1995; Friedrich et al., 2023; Y. Choi et al., 2024). Following X. Shen et al. (2024), we target a distribution of 75% Young and 25% Old for the age attribute. This choice facilitates distinct conclusions about each debiasing method, as illustrated in subsequent sections. It is important to note that the chosen attributes and classes reflect the constraints of the employed classifiers rather than limitations of the method itself (see Section 5.2). All experiments were conducted on a distributed setup of 10 NVIDIA A40 GPUs using half-precision floating-point format.

5.4.1 Standard Occupational Prompts

Experimental Setup The initial experiment assesses each method's overall effectiveness and general performance, constituting the most extensive evaluation in this study. We employ the prompt template *"A photo of the face of a <o>"* using a subset of 500 occupations from X. Shen et al. (2024). This prompt template and occupation list were used in the finetuning of the FDM model. To ensure fair comparison, we selected the same list of occupations used in training the h-space classifiers (see Section 4.3), albeit with a different prompt template. The inclusion of *"face of..."* in the template aims to enhance the performance of the employed face detection and attribute classification models. We generate 128 images per occupation using consistent seeds across all methods, resulting in 64,000 images per model and debiasing setup, totaling over one million generated images for this experiment. Generation was divided into batches of 64 images each, except for Fair Diffusion, which required a reduced batch size of 32 for debias setups with multiple attributes due to increased memory requirements.

General Performance We begin by presenting general results obtained during creation and evaluation of generated images. Pertaining to the overall utility, we illustrate each method's average time per image and memory consumption in Figure 11. As anticipated, the training time method (FDM) exhibits identical complexity during generation as the vanilla model, thus performing optimally among all methods in this regard. Attribute Switching follows closely, with slightly higher variances in time per image and memory consumption. Fair Diffusion demonstrates the poorest performance and scalability in both aspects. Comparing the GxRxA setup to the gender setup, time per image more than doubles while average memory consumption increases by approximately 25%.[6] Debias Diffusion's complexity is substantially lower than that of Fair Diffusion but slightly higher than the original model, with time complexity increased by 3-10% and memory requirements by 10-13% compared to the base model.

Fairness Table 1 reports the number of detected faces per method and the original model. The overall variance is relatively small, with each method generating strictly more faces than the vanilla model. Of the 500 employed occupations, 463 (92.6%) were correctly identified as human-related through linguistic analysis of the prompt using spaCy and NLTK (see Section 4.4.1). Images generated for occupations missed by this process resulted in a below-average number of faces. Consequently, 96% of all generated faces were successfully captured for debiasing purposes using the proposed approach. Fair Diffusion had a slight advantage among deployment stage methods in this regard, since it does not depend on these tools and was applied consistently across all prompts.

Further examining Table 1 reveals that all methods effectively mitigate bias and improve the model's fairness across all setups. Considering the average bias across all debiasing setups and attributes, Debias Diffusion consistently performs best, with minor exceptions in gender bias for the gender setup and age bias in the GxRxA setup, where it closely ranks second. The method demonstrates high effectiveness in simultaneously debiasing multiple

[6] Relative differences in memory consumption are less pronounced as the reserved memory for the model itself is included in these figures.

Debias Setup:	Method	Bias ↓			Detected Faces
		Gender	Race	Age	
—	Vanilla SD	.052±.167	.296±.176	.000±.136	58135
Gender	FDM	.026±.153	.270±.179	.002±.134	58526
	Fair Diff.	.005±.065	.254±.159	.004±.088	58340
	Attr. Swit.	.015±.103	.274±.173	.012±.084	58598
	Debias Diff.	.008±.118	.275±.172	.019±.075	58724
Race	FDM	.041±.165	.065±.108	.008±.134	59096
	Fair Diff.	.057±.149	.036±.045	.016±.050	59242
	Attr. Swit.	.058±.163	.014±.049	.001±.106	58374
	Debias Diff.	.065±.159	.006±.050	.000±.106	58379
G.×R.	FDM	.048±.145	.109±.120	.006±.138	59561
	Fair Diff.	.011±.068	.032±.029	.019±.043	59380
	Attr. Swit.	.018±.097	.002±.021	.021±.054	58801
	Debias Diff.	.009±.110	.002±.035	.015±.054	59204
G.×R.×A.	FDM	.019±.131	.104±.112	.000±.094	58398
	Fair Diff.	.019±.077	.029±.028	.011±.021	58611
	Attr. Swit.	.018±.092	.003±.026	.012±.027	58887
	Debias Diff.	.010±.108	.002±.038	.001±.064	59279

Table 1: Number of detected faces as well as bias values as measured by the FD metric for each debiasing method and combination of sensitive attributes. The lowest and therefore best bias value for each setup and sensitive attribute is marked in red. Additionally, the standard deviations across occupations are listed in grey.

attributes, with gender fairness deteriorating only minimally for more attributes, while race bias surprisingly decreases for multiple attributes. Notably, Attribute Switching and Debias Diffusion consistently achieve lower bias for race when it is a debiased attribute compared to the other two methods. FDM consistently struggles the most with improving fairness for debiased attributes, except for age bias in the GxRxA setup. However, the vanilla generative process was already fair on average regarding the target distribution for the age attribute. In this regard, we conclude that FDM performs best at maintaining the model's fair distribution while simultaneously debiasing the other two attributes, closely followed by Debias Diffusion.

Pertaining to group fairness (see also Section 5.3), all methods significantly lower the variance and thus improve group fairness with respect to all debiased attributes and setups. As depicted in Table 1, Fair Diffusion mostly achieves the lowest variance and therefore best group fairness, except for setups with multiple attributes where Attribute Switching achieves lower variances for race. Interestingly, Fair Diffusion also yields the best group fairness in almost all cases for attributes not actively being debiased. FDM consistently struggles the most with improving group fairness. Comparing both prompt editing methods, Attribute Switching achieves better group fairness for all debiased attributes than Debias Diffusion, while the opposite is true for attributes not included in any given debias setup.

For a more detailed analysis, we investigate the average and group fairness for race in the

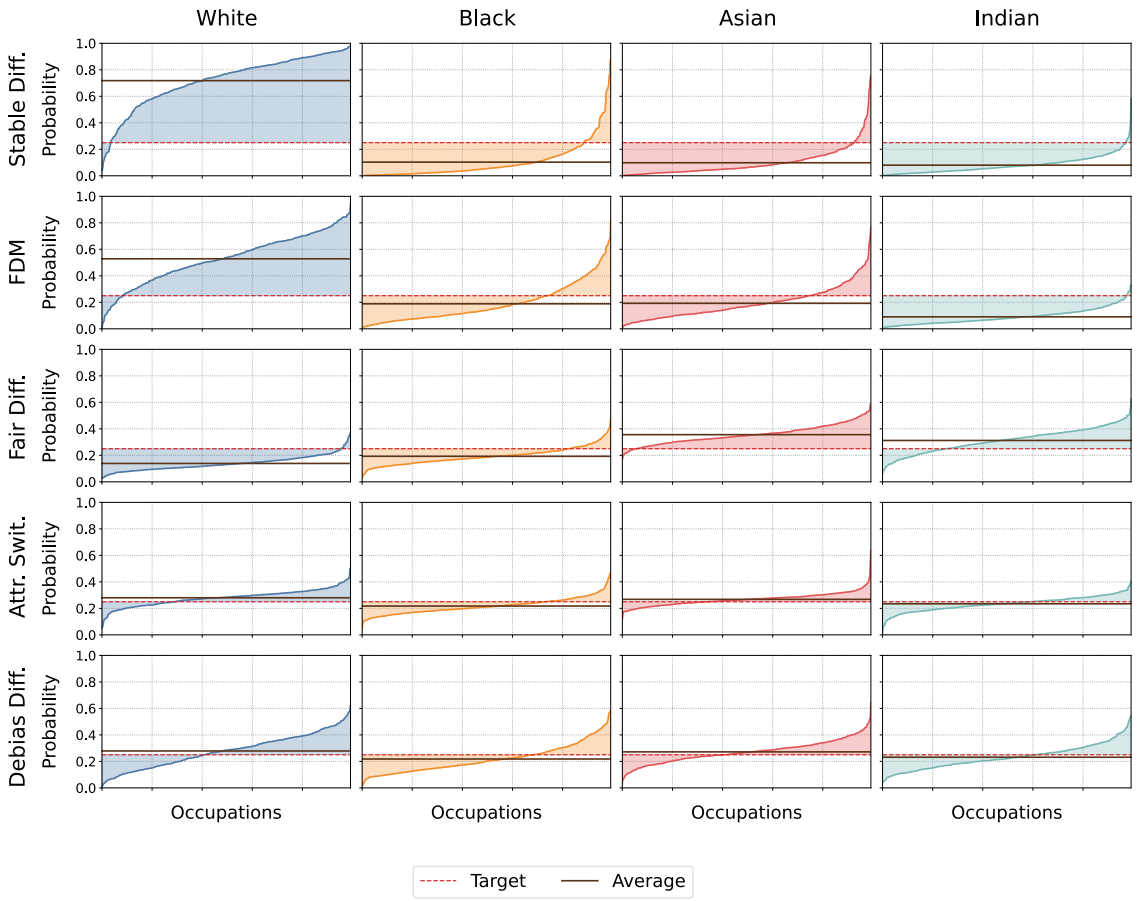


Figure 12: Race class probabilities over all occupations for each model in the race debiasing setup. The dotted red line indicates the target probability, and the brown line represents the average probability over all occupations.

race setup on a per-class level in Figure 12. All methods struggle to balance the majority class (White), with most approaches (except Fair Diffusion) still oversampling it. Debias Diffusion best mitigates the majority class bias. Similarly, all methods still under-represent the Black class, with FDM performing best in this regard. Except for Fair Diffusion, all methods manage to balance the Asian class, and, except for FDM, all models effectively debias the Indian class. FDM exhibits the most difficulty in mitigating extreme biases in single groups, as evidenced by the distribution's high peaks. Concerning the prompt editing methods, Debias Diffusion performs slightly worse for Asian and Indian classes as compared to Attribute Switching but better for the minority Black class and majority White class. Regarding fairness among gender and age classes, any present biases are due to oversampling of the majority classes, Male and Young, respectively.

Semantics Preservation Table 2 presents a comprehensive analysis of each method's performance in preserving original semantics. The FDM method consistently demonstrates superior text adherence while even surpassing the base model in most setups. Fair Diffusion generally ranks second, followed by Debias Diffusion and Attribute Switching,

Debias Setup:	Method	Semantics Preservation ↑			Quality ↓	
		CLIP-T	CLIP-I	LPIPS	FD_∞	KD
—	Vanilla SD	.708±.026	—	—	1107*	1.662*±.046
Gender	FDM	.713±.025	.875±.123	.740±.163	1150	1.720±.049
	Fair Diff.	.703±.026	.873±.113	.868±.080	1099	1.655±.045
	Attr. Swit.	.700±.026	.941±.074	.932±.060	1097	1.621±.045
	Debias Diff.	.703±.027	.920±.090	.867±.099	1103	1.631±.047
Race	FDM	.710±.026	.839±.113	.671±.160	1149	1.742±.047
	Fair Diff.	.698±.027	.845±.091	.833±.088	984	1.477±.042
	Attr. Swit.	.695±.027	.904±.083	.916±.059	996	1.450±.042
	Debias Diff.	.695±.027	.832±.113	.795±.104	962	1.397±.039
G.×R.	FDM	.710±.025	.812±.134	.619±.164	1167	1.687±.048
	Fair Diff.	.693±.027	.813±.100	.809±.085	986	1.488±.043
	Attr. Swit.	.688±.027	.861±.099	.890±.063	971	1.440±.040
	Debias Diff.	.688±.028	.813±.120	.775±.103	955	1.400±.033
G.×R.×A.	FDM	.708±.026	.804±.141	.618±.163	1184	1.721±.056
	Fair Diff.	.690±.028	.795±.105	.799±.079	985	1.470±.038
	Attr. Swit.	.685±.027	.843±.104	.874±.068	949	1.402±.043
	Debias Diff.	.685±.028	.801±.121	.764±.103	940	1.353±.038

Table 2: Semantics preservation and quality metrics as measured by each metric for all debiasing method and setup. Favorable trends are indicated by arrows and best values in each debiasing setup marked in red. Additionally, the standard deviations across occupations are listed in grey.

which exhibit remarkably similar performance in terms of text adherence. Notably, all methods show a decline in performance as the number of classes and attributes increases among setups.

In terms of image similarity to original generations, Attribute Switching distinctly outperforms other methods for both CLIP-I and LPIPS metrics. The remaining three methods show comparable performance for CLIP-I, with Debias Diffusion nearly matching Attribute Switching in the gender setup. Interestingly, the FDM model consistently underperforms in LPIPS metrics compared to its CLIP-I scores. A common trend observed across all methods is the degradation of performance in both metrics as the complexity of classes and attributes increases.

Quality Regarding overall image quality in Table 2, Debias Diffusion excels in almost all debiasing setups, ranking a close second only in the gender category. FDM consistently lags behind by a significant margin according to both metrics, with its performance further deteriorating for multiple attributes as per the FD_∞ metric. Conversely, Attribute Switching and Debias Diffusion show consistent improvement in both metrics across most setups as the number of classes and attributes increases.

To further evaluate each method's visual characteristics and perceptual quality, an extensive range of comparative example images were analyzed. Figure 13 illustrate such an

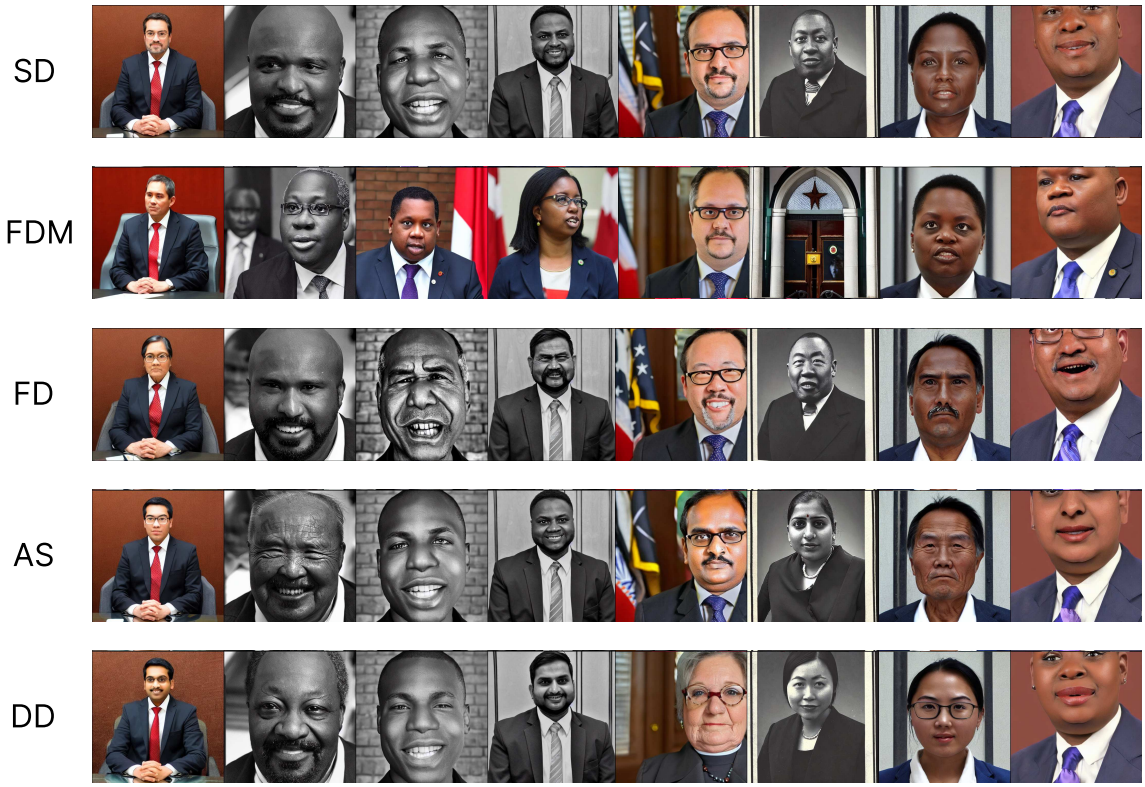


Figure 13: Generated results for the prompt *"A photo of the face of a minister"* for each method in the GxRxA debiasing setup. Results for the same seeds were randomly selected from the evaluated datasets. The model exhibits a rather balanced distribution over all attributes for this domain with average FD values of 0.022 (age), 0.134 (race), and 0.034 (gender).

example for a highly biased domain in the GxRxA setup. All methods demonstrate high utility, producing images of comparable overall quality to the base model. Fair Diffusion exhibits the most noticeable artifacts, with images occasionally appearing "printed on" or featuring exaggerated characteristics. In contrast, FDM shows the highest stability in terms of image integrity, with minimal systematic artifacts. Both Attribute Switching and Debias Diffusion produce high-quality, utilitarian images. However, Attribute Switching occasionally generates examples that appear to blend characteristics from multiple classes within the same attribute. FDM, while stable, shows the greatest deviation from original image semantics. The perceptual quality, as assessed through a large collection of evaluated examples, aligns more closely with LPIPS than CLIP-I metrics, as indicated by the metrics name (see Section 5.3.2). Additional examples for various occupations are provided in the Appendix, offering a more comprehensive insight into each method's exhibited qualities.

Debias Setup:	Method	Bias ↓			Detected Faces
		Gender	Race	Age	
—	Vanilla SD	.032±.136	.218±.180	.000±.141	8928
Gender	FDM	.027±.131	.223±.183	.001±.121	8962
	Fair Diff.	.010±.075	.204±.164	.003±.091	8953
	Attr. Swit.	.024±.110	.215±.175	.006±.087	9023
	Debias Diff.	.017±.128	.215±.177	.008±.090	9100
Race	FDM	.028±.137	.175±.163	.002±.110	8985
	Fair Diff.	.026±.123	.063±.043	.005±.059	8957
	Attr. Swit.	.035±.136	.071±.073	.003±.109	9031
	Debias Diff.	.034±.137	.049±.070	.002±.096	9192
G.×R.	FDM	.027±.136	.191±.164	.002±.129	8987
	Fair Diff.	.013±.063	.055±.041	.006±.044	8927
	Attr. Swit.	.022±.109	.047±.060	.010±.072	9191
	Debias Diff.	.013±.124	.045±.066	.009±.069	9346
G.×R.×A.	FDM	.023±.133	.166±.149	.001±.129	8958
	Fair Diff.	.013±.069	.059±.040	.005±.048	8935
	Attr. Swit.	.019±.107	.051±.057	.012±.040	9199
	Debias Diff.	.018±.126	.054±.057	.005±.047	9432

Table 3: Number of detected faces as well as bias values as measured by the FD metric for each debiasing method and combination of sensitive attributes. The lowest and therefore best bias value for each setup and sensitive attribute is marked in red. Additionally, the standard deviations across prompts are listed in grey.

5.4.2 LAION-Aesthetics Prompts

Experimental Setup The second experiment broadens the analysis to non-templated prompts, utilizing a set of 40 occupation-related prompts employed by X. Shen et al. (2024), derived from the *LAION-Aesthetics V2* dataset (Schuhmann et al., 2022). These prompts are non-templated and more complex than previous ones. They encompass diverse descriptions, such as *"Attractive steampunk singer with microphone"* and *"A philosopher reading. Oil painting."*. For each prompt, 256 images are generated, totaling 10,240 images per method and debiasing setup which amounts to well over 170.000 images analyzed in this experiment.

Fairness As shown in Table 3, almost all methods generate more faces than the base model across all setups, with Debias Diffusion consistently producing the highest number. Fair Diffusion exhibits the least variance from the original base model's face count. For the prompt modification methods (Debias Diffusion and Attribute Switching), linguistic analysis successfully identified human-related nouns in 38 out of 40 prompts (95%). The missed prompts resulted in a below-average number of faces, leading to syntactic filtering successfully capturing 96.6% of all generated faces for debiasing purposes.

Examination of the bias metrics in Table 3 reveals improved fairness for debiased attributes across all methods as well as several repeating patterns observed in the first experiment

Debias Setup:	Method	Semantics Preservation ↑			Quality ↓	
		CLIP-T	CLIP-I	LPIPS	FD_∞	KD
—	Vanilla SD	.800±.033	—	—	1309	1.246±.032
Gender	FDM	.800±.034	.931±.074	.808±.132	1335	1.262±.031
	Fair Diff.	.790±.033	.927±.062	.868±.072	1285	1.235±.035
	Attr. Swit.	.790±.033	.950±.058	.928±.053	1283	1.242±.031
	Debias Diff.	.788±.033	.921±.075	.844±.096	1291	1.256±.031
Race	FDM	.800±.034	.915±.080	.764±.136	1410	1.396±.035
	Fair Diff.	.775±.034	.869±.084	.809±.087	1171	1.170±.034
	Attr. Swit.	.790±.034	.937±.069	.920±.054	1292	1.294±.034
	Debias Diff.	.778±.035	.854±.104	.721±.120	1202	1.221±.030
G.×R.	FDM	.798±.034	.904±.083	.732±.138	1378	1.337±.035
	Fair Diff.	.765±.034	.848±.092	.789±.086	1139	1.119±.035
	Attr. Swit.	.780±.034	.904±.088	.895±.061	1220	1.208±.033
	Debias Diff.	.768±.035	.832±.112	.710±.119	1156	1.164±.030
G.×R.×A.	FDM	.798±.034	.899±.090	.729±.142	1129	1.111±.026
	Fair Diff.	.765±.034	.849±.091	.790±.086	1124	1.117±.031
	Attr. Swit.	.778±.035	.897±.092	.886±.063	1197	1.186±.030
	Debias Diff.	.765±.036	.842±.114	.750±.118	1135	1.141±.033

Table 4: Semantics preservation and quality metrics as measured by each metric for all debiasing method and setup. Favorable trends are indicated by arrows and best values in each debiasing setup marked in red. Additionally, the standard deviations across prompts are listed in grey.

(Section 5.4.1). Notably, the base model's age distribution again aligns with the target distribution. As in the first experiment, Debias Diffusion consistently performs best or second-best across all debiased attributes in any given setup. Fair Diffusion generally achieves the best results for gender bias across all setups even if it is not actively debiased. FDM consistently underperforms when debiasing for race or age (or both), with its performance on race being particularly weak compared to other methods and previous results. However, similar to the first experiment, it again demonstrates the best preservation of the age distribution.

Regarding group fairness, all methods either maintain or improve fairness for all debiased attributes. Fair Diffusion consistently outperforms in this aspect for all debiased attributes and exhibits the least variance among methods, even for attributes not included in any given debiasing setup.

Individual class analysis reveals that most methods still struggle with oversampling majority classes across all attributes. The only exception being Fair Diffusion which manages to balance the majority class for race (White) when race is a debiased attribute. The Asian class, initially well-balanced in the base model, shows imbalance across all methods and setups. Black and Indian classes remain under-sampled in all methods and setups, though generally showing improvement over the original model.

Figure 14: Generated results for the prompt *"English writer and essayist"* for each method in the GxRxA debiasing setup. Results for the same seeds were randomly selected from the evaluated datasets. The model exhibits a strong bias towards Old, White, and Male classes with average FD values of 0.632 (age), 0.665 (race), and 0.337 (gender).

Semantics Preservation Table 4 demonstrates that the original model and FDM exhibit the best prompt adherence, with FDM performing consistently well across all setups. Attribute Switching ranks second, while Fair Diffusion and Debias Diffusion are closely tied for last. All methods show declining performance as more classes and attributes are incorporated into the debiasing setup.

In terms of image semantics preservation, Attribute Switching generally outperforms in most setups for both metrics, though the discrepancies are less pronounced than in the previous experiment. Debias Diffusion introduces the most semantic change across all debiasing setups according to CLIP, and the most or second most according to LPIPS. Very similar to previous observations, FDM once again yields notably higher CLIP-I than LPIPS values. This becomes particularly evident when comparing Attribute Switching with FDM, as both have similar CLIP-I values but differ significantly in LPIPS rankings. It is worth noting that the values for text adherence and image similarity are very similar for FDM and Attribute Switching, as well as for Fair Diffusion and Debias Diffusion with some minor exceptions.

Quality Analysis of the quality metrics in Table 4 shows that Fair Diffusion demonstrates superior performance in all setups for FD_∞, except gender, and for KD, except for

the GxRxA setup. While the correlation between both metrics is still evident, it is less pronounced than in the previous experiment. FDM shows notably improved performance in the GxRxA setup compared to other setups and its values from the previous experiment. Debias Diffusion consistently ranks second or third in both metrics. To illustrate how each method handles extreme cases, Figure 14 presents generations in the GxRxA setup for a domain highly biased across all three attributes.

5.4.3 Real-World User Prompts

Experimental Setup This experiment aims to complement previous evaluations by assessing even more complex, realistic, and non-occupation-related prompts. To this end, prompts collected by X. Shen et al. (2024) from the currently largest available open-source TTI prompt dataset, *DiffusionDB* (Z. J. Wang et al., 2023), are utilized. This dataset comprises web-scraped images and corresponding prompts employed by real users in the official Stable Diffusion Discord server (StabilityAI, 2024a). Initially, the complete list of 150 prompts from X. Shen et al. (2024) was considered. However, upon manual examination, 12 out of 13 prompts not recognized for debiasing by NLTK proved to be instances of known entities, e.g., *"judge judy"* or *"captain picard"*. As these are intentionally excluded from debiasing by design (see Section 4.4.1), the evaluation focuses on the remaining 138 prompts. Examples include *"a farmer by andy warhol, digital art, trending on artstation"* and *"painting of a deploying soldier from a helicopter in the gulf war by Bernardo Bellotto, high detail, hyperrealistic, concept art, artstation, 8k"*. For each prompt, 128 images are generated, yielding 17,664 images per method and setup, totaling over 300,000 images for the entire experiment.

Fairness The fairness analysis begins by assessing the number of generated faces across all methods. Consistent with previous experiments, all methods yield more faces than the base model in almost all setups. Notably, Fair Diffusion creates significantly more images depicting human faces compared to other methods and in relation to previous experiments. Regarding syntactic filtering, only one prompt was not correctly filtered for human-related nouns, resulting in 99.3% of prompts and faces being considered for debiasing purposes.

Concerning average fairness across setups, all methods successfully mitigate bias for all debiased attributes under consideration. It is worth noting that age is not quite perfectly aligned with the target distribution for the base model in this experiment. Interestingly, almost all methods perform better with respect to debiasing age than in previous experiments. Similar to both previous experiments, Debias Diffusion ranks as best or second best in terms of achieved fairness for all debiased attributes. Fair Diffusion performs best for race in all setups, while FDM notably struggles with mitigating race bias. Furthermore, both prompt editing methods show decreased performance in debiasing race compared to previous experiments.

Examining group fairness within Table 5, it is notable that the variances for gender in all setups are the highest across all experiments. Fair Diffusion, in particular, does not improve group fairness for attributes gender and age as effectively as in previous examples. Concerning group fairness within the race attribute, both prompt editing methods yield higher variances than previously observed.

Debias Setup:	Method	Bias ↓			Detected Faces
		Gender	Race	Age	
—	Vanilla SD	.018±.145	.200±.158	.003±.190	14295
Gender	FDM	.009±.145	.205±.160	.000±.185	14374
	Fair Diff.	.011±.118	.197±.150	.002±.165	15576
	Attr. Swit.	.010±.133	.209±.156	.000±.175	14659
	Debias Diff.	.004±.130	.221±.159	.000±.166	14880
Race	FDM	.019±.144	.163±.158	.000±.185	14657
	Fair Diff.	.020±.137	.046±.072	.000±.144	15456
	Attr. Swit.	.020±.148	.117±.140	.001±.179	14509
	Debias Diff.	.021±.147	.107±.139	.001±.184	14533
G.×R.	FDM	.011±.144	.163±.142	.002±.184	13909
	Fair Diff.	.013±.107	.042±.062	.001±.132	15549
	Attr. Swit.	.011±.132	.092±.130	.000±.156	14848
	Debias Diff.	.006±.130	.080±.124	.002±.141	15191
G.×R.×A.	FDM	.015±.143	.176±.149	.001±.181	14326
	Fair Diff.	.010±.109	.036±.063	.009±.110	15051
	Attr. Swit.	.012±.136	.087±.124	.001±.141	15032
	Debias Diff.	.007±.129	.084±.126	.001±.144	15212

Table 5: Number of detected faces as well as bias values as measured by the FD metric for each debiasing method and combination of sensitive attributes. The lowest and therefore best bias value for each setup and sensitive attribute is marked in red. Additionally, the standard deviations across prompts are listed in grey.

Lastly, assessing fairness at the individual class level reveals that all methods continue to oversample majority classes for gender (Male) and race (White). However, Fair Diffusion and the base model itself undersample the majority class for age (Young). Regarding race specifically, all methods again undersample the minority class Black, though more pronouncedly than in previous examples.

Semantics Preservation An analysis of Table 6 reveals that, for the first time across all three experiments, the original model demonstrates superior performance in prompt adherence. FDM follows closely, with Attribute Switching trailing by a narrow margin, except in the GxRxA setup where the gap widens noticeably. Debias Diffusion maintains a competitive third position, while Fair Diffusion lags significantly, particularly in debiasing setups involving multiple attributes.

Regarding image similarity to the original model's outputs, this experiment exhibits the most pronounced disparities between CLIP-I and LPIPS metrics observed thus far. Attribute Switching stands out as an exception, consistently achieving the highest scores across both metrics and all setups. Notably, the values for these metrics no longer display the near-linear decline previously observed when additional classes and attributes were incorporated into a given setup. It is worth highlighting that Fair Diffusion encounters challenges with image similarity in the gender setup according to LPIPS.

Debias Setup:	Method	Semantics Preservation ↑			Quality ↓	
		CLIP-T	CLIP-I	LPIPS	FD_∞	KD
—	Vanilla SD	.828±.038	—	—	1517	2.741±.055
Gender	FDM	.823±.038	.934±.057	.794±.117	1528	2.779±.059
	Fair Diff.	.815±.038	.886±.090	.663±.264	1539	2.811±.054
	Attr. Swit.	.823±.038	.970±.039	.937±.055	1531	2.827±.055
	Debias Diff.	.818±.038	.917±.065	.789±.105	1537	2.892±.057
Race	FDM	.823±.038	.920±.060	.754±.118	1620	3.011±.056
	Fair Diff.	.800±.038	.897±.063	.791±.087	1434	2.686±.056
	Attr. Swit.	.820±.038	.971±.036	.931±.054	1539	2.832±.066
	Debias Diff.	.815±.038	.942±.052	.851±.088	1525	2.814±.056
G.×R.	FDM	.815±.038	.905±.066	.720±.121	1595	2.922±.035
	Fair Diff.	.775±.038	.877±.070	.769±.086	1399	2.635±.050
	Attr. Swit.	.813±.038	.950±.055	.908±.066	1505	2.792±.061
	Debias Diff.	.803±.039	.896±.076	.772±.102	1485	2.798±.058
G.×R.×A.	FDM	.823±.039	.910±.065	.732±.118	1579	2.892±.067
	Fair Diff.	.785±.038	.879±.070	.764±.084	1495	2.940±.059
	Attr. Swit.	.810±.038	.944±.059	.898±.068	1478	2.738±.051
	Debias Diff.	.803±.039	.893±.077	.771±.102	1471	2.730±.063

Table 6: Semantics preservation and quality metrics as measured by each metric for all debiasing method and setup. Favorable trends are indicated by arrows and best values in each debiasing setup marked in red. Additionally, the standard deviations across prompts are listed in grey.

Quality Analysis of the quality metrics presented in Table 6 reveals less distinct trends or patterns compared to previous experiments. Notably, Debias Diffusion once again demonstrates consistent improvement in both metrics as more classes and attributes are incorporated into any given debiasing setup. FDM, in line with earlier observations, generally exhibits the poorest performance, except for the gender setup where it ranks as first. Lastly, the previously documented high correlation between both metrics is also reflected in this experiment.

5.5 Discussion

This section discusses the evaluation results, focusing on the key factors outlined in Section 4.1: complexity, performance, and usability.

Complexity Debias Diffusion introduces a modest increase in complexity compared to the base model, primarily due to the additional h-space classifiers. The syntactic filtering process proves to be highly efficient, as evidenced by the minimal difference in time and memory complexity between the original method and Attribute Switching (Figure 11). The classifiers themselves are lightweight and fast relative to the UNet, which remains the primary source of computational requirements. The majority of added time complexity in Debias Diffusion stems from the reversion of the generative process in cases of high esti-

mated bias. While the pre-deployment training of h-space classifiers must be considered, it is significantly less resource-intensive than, for example, the finetuning of the FDM model, and is achievable with consumer-grade hardware (see Section 5.1.1). Moreover, the proposed alternative training methods offer flexibility in training classifiers for various attributes with minimal external dependencies. Fair Diffusion emerges as the most resource-intensive method due to its reliance on UNet predictions for each encoded concept or class. Therefore, it significantly outweighs the combined cost of syntactic filtering, h-space predictions, and prompt encodings.

Performance The empirical results effectively address the concerns raised by Friedrich et al. (2023) regarding the efficacy of linguistic prompt modification for debiasing. Debias Diffusion demonstrates high competitiveness in fairness, achieving the lowest bias for 12 out of 21 debiased attributes across all experiments. It consistently outperforms Attribute Switching in 19 of these attributes, albeit at the cost of reduced preservation of original image semantics. This trade-off can be attributed to various factors influencing the dynamic selection of insertion timepoints, as elaborated in Section 6.1.2. The theory presented in Section 4.2 suggests that a fixed insertion step towards the end of the content phase, as employed by Attribute Switching, should result in less overall semantic change than insertion steps earlier in the same phase. Thus, the hypothesis emerges that greater semantic change is necessary to approach fairness more closely. Debias Diffusion's dynamic mechanism for detecting suitable insertion timesteps and optimal target classes offers an informed balance between semantic preservation and fairness. As demonstrated in Section 6.1.2, this balance is additionally influenced by the hyperparameters batch size and τ_{bias}.

A comprehensive examination of example image grids across all experiments indicates that all methods produce utilitarian results of perceptual quality comparable to the original model. While acknowledging the potential subjectivity of such evaluations, it is noteworthy that both prompt editing methods yielded fewer artifacts and often more natural-appearing results compared to Fair Diffusion. Although different hyperparameter choices for Fair Diffusion might yield improved results, the ability to avoid domain-specific finetuning while still achieving highly debiased and utilitarian results is a feature enabled by the performant syntactic approach presented in Section 4.4.

The lower prompt adherence observed for Debias Diffusion and Attribute Switching compared to FDM and the original model was anticipated, given that both methods effectively alter the prompt itself. Notably, while these methods perform similarly in the first experiment, Attribute Switching outperforms Debias Diffusion regarding prompt adherence in subsequent experiments, with the margin growing for more classes and attributes. This trend can likely be attributed to the increased semantic change introduced by Debias Diffusion in pursuit of fairness.

Usability In terms of flexibility to adapt to different target distributions and attributes, FDM presents the least favorable characteristics among the evaluated methods. It necessitates expensive finetuning and auxiliary requirements (datasets, classification models, etc.) for each new attribute and target distribution, effectively precluding its application using consumer hardware. Conversely, Fair Diffusion offers the most flexibility and sim-

plicity regarding auxiliary requirements and additional prior work. However, its utility is significantly compromised by poor scaling properties in terms of time and memory with respect to new classes and attributes.

The prompt editing methods, Attribute Switching and Debias Diffusion, require the availability of NLTK's WordNet and spaCy's models, which represents a minor added complexity. This approach allows for dynamic addition of new attributes or changes to target distributions between generations. The technique can be further enhanced, for example, by incorporating synonymous concepts for each class to insert into prompts, which we leave as possible further research. Debias Diffusion shares these qualities while achieving improved fairness through dynamic insertion point selection. This improvement comes at the cost of slightly decreased utility due to the prior training of h-space classifiers for each attribute. However, these classifiers are lightweight and can be trained relatively quickly on consumer hardware (see Section 5.1.1).

Lastly, we again emphasize the consistent discrepancy between CLIP-I and LPIPS values for FDM across all experiments and setups. This may be attributed to an over-reliance on CLIP in the loss design, despite the authors' explicit efforts to mitigate such effects by incorporating signals from the DINO model. This discrepancy, not addressed in the original work, presents an interesting avenue for future research.

6 Ablation Studies

The following ablation studies examine various hyperparameters and components of the presented method, Debias Diffusion, as well as the overall accuracy of the h-space classifiers.

6.1 Debias Diffusion

Subsequent experiments aim to elucidate the impact of different design choices with respect to the proposed method (see Section 4.4) and provide insights into potential areas for improvement. Section 6.1.2 will additionally uncover some unexpected properties of the trained h-space classifiers.

6.1.1 False Positive Rate of Syntactic Filtering

The efficacy of the syntactic filtering mechanism described in Section 4.4.1 was empirically validated through three comprehensive experiments detailed in Section 5. Conceptually, this filtering process can be viewed as a classification task performed by NLTK on candidate words identified by spaCy. To further validate this approach, we aimed to measure its false-positive rate. To this end, we required a substantial dataset of prompts unlikely to contain content related to human subject depiction. After considering various options, we sampled the LAION-400m dataset to collect 1,000 images and their corresponding prompts or captions, ensuring that none of the images contained human faces. Representative examples include prompts such as *"Snow on Main Street. 11/7/12"* and *"San Jorge Golf Club Logo"*.Upon application of the syntactic filtering mechanism, 799 prompts (79.9%) were correctly identified as not human-related. A subsequent manual inspection of the remaining 201 prompts revealed that 21 did, in fact, contain human-related nouns (e.g., *"worker on fig plantation"*). After accounting for these instances, the rate of incorrectly flagged prompts decreased to 180 out of 1,000 (18%). To assess the impact of this false-positive rate on image generation, we produced batches of 32 images for each of the 1,000 prompts using both the original Stable Diffusion model and Debias Diffusion. The resulting text adherence was identical for both methods. Moreover, the high CLIP-I (.970) and LPIPS (.936) values indicate that Debias Diffusion successfully preserves original semantics in domains that do not require debiasing.

6.1.2 Batch Size and τ_{bias}

To investigate the influence of batch size and τ_{bias} (the timestep at which bias is measured to compute insertion points), a subset of 100 occupations from the list used in Section 5.4.1 is employed. Using the same prompt template, 128 images are generated for each occupation. The experiment utilizes varying batch sizes corresponding to constraints of common commercial and data-center GPUs: 16 for RTX A4000/V100 (16GB VRAM), 32 for RTX A5000/A30 (24GB), 64 for RTX A6000/A40 (48GB), and 128 for A100 (80GB). Additionally, four timesteps evenly spaced across the model's content stage are considered: $\tau_{bias} \in \{46, 41, 36, 31\}$. This setup results in 12,800 images for each combination of batch

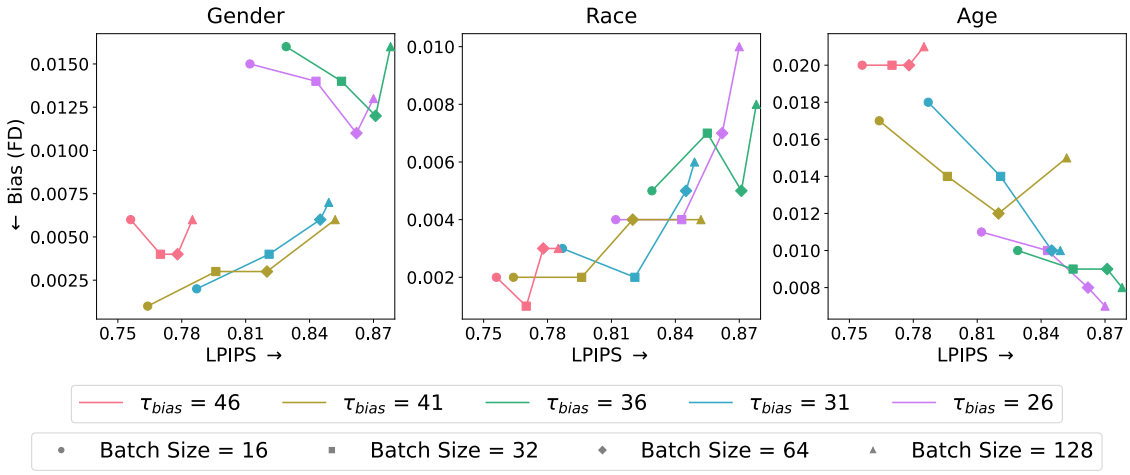


Figure 15: Relationship between bias and semantic preservation for different batch sizes and τ_{bias} values.

size and τ_{bias}, with debiasing applied to race and gender attributes simultaneously using balanced target distributions for each.

The analysis of these hyperparameters is motivated by their potential impact on the method's performance. Larger batch sizes are expected to benefit both the optimal transport (OT) problem solution and the estimated bias calculation. The OT problem can be solved with more degrees of freedom and a larger sample size for computing the expectation of the estimated bias constitutes a less noisy approximation. Assuming reasonably accurate h-space classifiers, an improved OT solution should contribute to better preservation of original semantics, while more accurate bias estimation should lead to improved class adherence and overall fairness. The choice of τ_{bias} directly determines the classifier used to predict batch element classes during generation. Theoretical considerations from Section 4.2 suggest that classifiers become more accurate for smaller timesteps later in the generative process, as latent trajectories converge towards class-related features. Initially, it was expected that inaccurate classifiers in high timestep regimes would yield an approximately equal probability mass distribution over classes due to uncertainty and consequently low biases. This would result in concentration of timesteps towards the end of the content phase, approximating the Attribute Switching method (see also Section 5.1.3). Assuming that the optimal insertion timepoints for class modifiers indeed depend on the model's biases as hypothesized, Debias Diffusion should perform better as bias estimations become more accurate. However, the results will demonstrate that this expectation is only partially met due to unexpected characteristics of the h-space classifiers.

While both hyperparameters influence time and memory efficiency, this experiment focuses on their impact on fairness (measured by average bias across occupations) and preservation of original semantics (measured by average LPIPS across generated samples). The relationship between these metrics is referred to as the *bias-SP trade-off*, where SP denotes semantic preservation. Figure 15 presents the results, plotting both metrics against each other. By design, the best outcomes are located near the lower right corner of each subplot. Datapoints with the same h-space classifier (same τ_{bias}) are connected as line

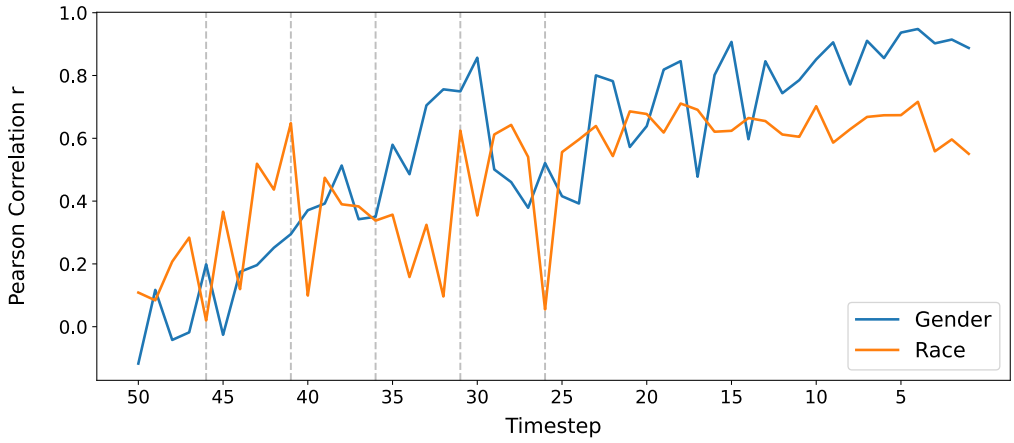


Figure 16: Correlation between estimated and real bias for different timesteps. Timesteps $\tau_{bias} \in \{46, 41, 36, 31\}$ are marked by gray, dotted lines due to their relevance for the analysis.

plots over increasing batch sizes, which in turn are grouped under the same marker. Key observations include:

1. Higher batch sizes consistently yield better SP across all attributes and choices of τ_{bias}, as evidenced by the monotonicity of each line plot along the x-axis.
2. For gender and race, higher batch sizes mostly yield higher bias, while the opposite is true for age.
3. Regarding gender, results for $\tau_{bias} = 46$ focus on high fairness and low SP, while $\tau_{bias} \in \{36, 26\}$ yield high SP and low fairness. No clear tendencies are observable for race with respect to different τ_{bias} values.

To explain these results, the accuracy of h-space classifiers is assessed by comparing the 'true' bias (estimated by high-quality image classifiers, see Section 5.2) with the estimated bias from h-space classifiers at each timestep for a batch size of 128. Figure 16 illustrates the correlation between these biases using the Pearson correlation coefficient for gender and race separately. The general trend of rising correlations confirms that classifiers indeed become more accurate for both attributes as generation progresses, although variance between close timesteps remains high. As such, we hypothesize that the first observation is mainly caused by the enhancement of the OT solution for larger batches. There is additional support for this assumption. For example by focusing on the results of $\tau_{bias} \in \{46, 31\}$ with regards to gender. These timesteps show the greatest difference in correlation of biases among the evaluated ones as evidenced in Figure 16. A comparison of their results for batch size 128 in Figure 15 reveals a significant increase in SP but only a slight decrease in fairness. This indicates that the method benefits from larger batch sizes when classifiers are reasonably accurate.

Contrary to expectations, fairness does not improve with growing batch sizes for debiased attributes (observation 2). To investigate this phenomenon further, the relationship

between real and estimated bias for each timestep and attribute is plotted in Appendix A. This analysis reveals that biases in early timesteps are highly concentrated on certain values or small ranges. As was revealed by further inspection, this phenomenon is caused by a tendency of h-space classifiers in early timesteps to focus probability mass on specific classes rather than encoding high variance to reflect uncertainty. For example, gender classifiers in the coarse stage display a pronounced bias towards the Female class, which increases until about timestep 48, after which probability mass gradually disperses. For race, the first timestep focuses almost all probability mass on the White class, with subsequent timesteps often concentrating on two out of four classes, which change between timesteps. According to the formula in Section 3.3, the bias for an equal distribution over two classes is exactly 0.25. As shown in Figure 20, many of the estimated biases in early timesteps cluster closely around this value. These characteristics of each timestep as observed in Appendix A explain most variations in fairness observed in Figure 15 and especially the third key observation. For example, the gender classifier for $\tau_{bias} = 46$ yields very high estimated bias values on average, leading to early insertion timesteps. These cause better class adherence, but very low SP. Additionally, the prolonged modification of the generative process introduces notable new bias with respect to age. This can likely be attributed to cross association inherent to the model itself between the debiased attributes and age. Conversely, classifiers for $\tau_{bias} \in \{36, 26\}$ yield low estimated bias values, causing opposite effects. The second observation is a consequence of the flawed initial assumption about h-space classifiers. Their tendency to be biased rather than 'random' with respect to class probabilities in early timesteps means that larger batches amplify these effects, leading to suboptimal insertion steps and consequently worse class adherence and fairness.

In summary, while the method demonstrates overall effectiveness in improving fairness compared to the base model (see also Section 5.4), the use of prompt-based interventions for debiasing has inherent limitations. These include difficulties in accounting for new biases introduced with respect to other attributes (e.g., age in this experiment) and apparent limits to improving the bias-SP trade-off through linguistic modifier insertion. However, the results indicate that a good balance in the bias-SP trade-off can be achieved using Debias Diffusion if the h-space classifier at the chosen τ_{bias} is sufficiently accurate. However, these classifiers can be surprisingly biased in early generation steps which may need special treatment in an improved method. Future research opportunities include focusing on training h-space classifiers for a single timestep τ_{bias}, exploring alternative metrics to the FD metric, and investigating different interpolation functions between bias and insertion timestep.

7 Conclusion

This thesis presents several key contributions to the field of fair generative modeling. The primary contribution is the development of Debias Diffusion (Section 4.4), a novel method for mitigating biases in text-to-image diffusion models. Additionally, we introduce a custom, lightweight training method for h-space classifiers (Section 4.3), and additionally reveal interesting properties regarding probability mass distribution of these classifiers in early diffusion timesteps (Section 6.1.2). Debias Diffusion focuses on linguistic interventions in prompts used for text-to-image models to achieve fairer outputs. Unlike many contemporary approaches, it utilizes NLP tools such as spaCy (Honnibal et al., 2020) and NLTK (Bird et al., 2009) to create a syntactic representation of prompts, enabling targeted insertion of linguistic modifiers. Our work therefore empirically addresses concerns raised by Friedrich et al. (2023) regarding the efficacy of textual interventions for debiasing purposes. Building on insights from Y. Choi et al. (2024), Debias Diffusion demonstrates that generations can be guided towards target classes through well-timed insertions of modifiers in the generative process. We empirically validate this by evaluation using diverse sets of prompts, generating well over 1.5 million images in total over four competing debiasing methods. As shown, Debias Diffusion offers an informed approach which balances the desired debiasing effect with the preservation of original semantic content.

Our findings suggest a correlation between optimal switching timesteps and the model's inherent bias in the domain of the relevant sensitive attribute, though ablation studies in Section 6.1.2 indicate potential limitations to this approach. Despite these constraints, Debias Diffusion performs competitively with, and often surpasses, state-of-the-art methods, as evidenced in Section 5. Notably, it achieves superior results to a method employing resource-intensive finetuning of model parameters (X. Shen et al., 2024). Furthermore, the presented dynamic approach to detecting optimal insertion timesteps in batch generation consistently outperforms the theoretically derived optimal switching points proposed by Y. Choi et al. (2024) in terms of debiasing efficacy.

A key advantage of Debias Diffusion is its flexibility and minimal external requirements. It allows for on-the-fly adjustments to target distributions and attributes between batch generations, complemented by the efficient training of h-space classifiers for various sensitive attributes of interest (Section 4.3). These classifiers demonstrate the ability to classify sensitive attributes with reasonable accuracy early in the generative process (Section 6.1.2). The overall method, Debias Diffusion, achieves competitive performance even when 'unfairly' compared to methods that omit any type of input filtering on evaluated prompts, further showcasing the effectiveness of our syntactic filtering approach (Section 4.4.1). This work opens several avenues for future research, including further exploration of the semantic latent space in diffusion models and refinement of linguistic intervention techniques for bias mitigation. The insights gained from this study contribute to the ongoing efforts to develop fair and unbiased generative models and might help understanding the cause of such biases, an increasingly critical area in the field of machine learning.

A Appendix

Estimated Bias vs Real Bias

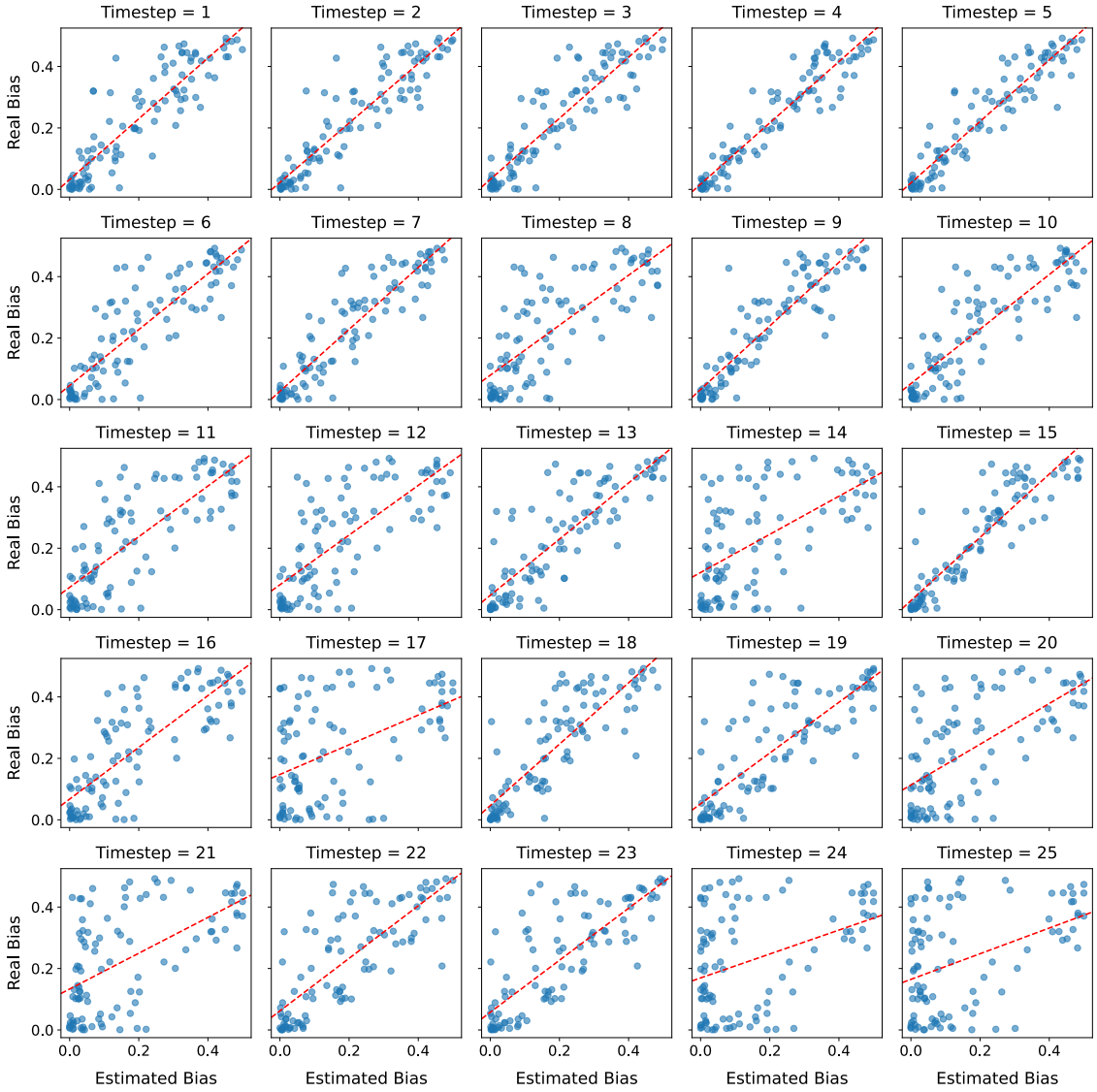


Figure 17: Comparison of real and estimated gender biases for 100 occupations (blue dots) as well as their correlation trend line in the G.x.R. debiasing setup (timesteps 25 to 1).

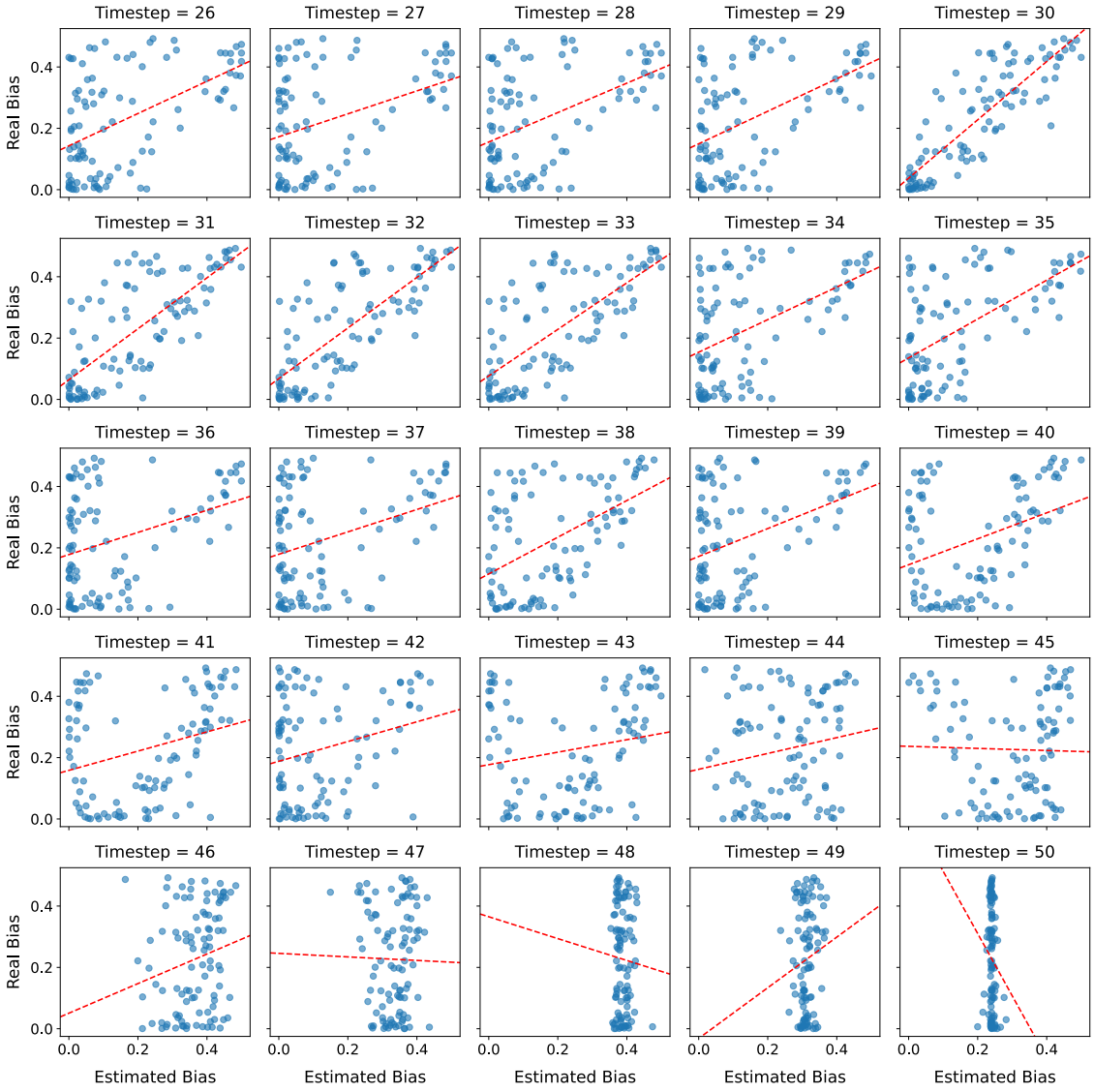


Figure 18: Comparison of real and estimated gender biases for 100 occupations (blue dots) as well as their correlation trend line in the G.x.R. debiasing setup (timesteps 50 to 26).

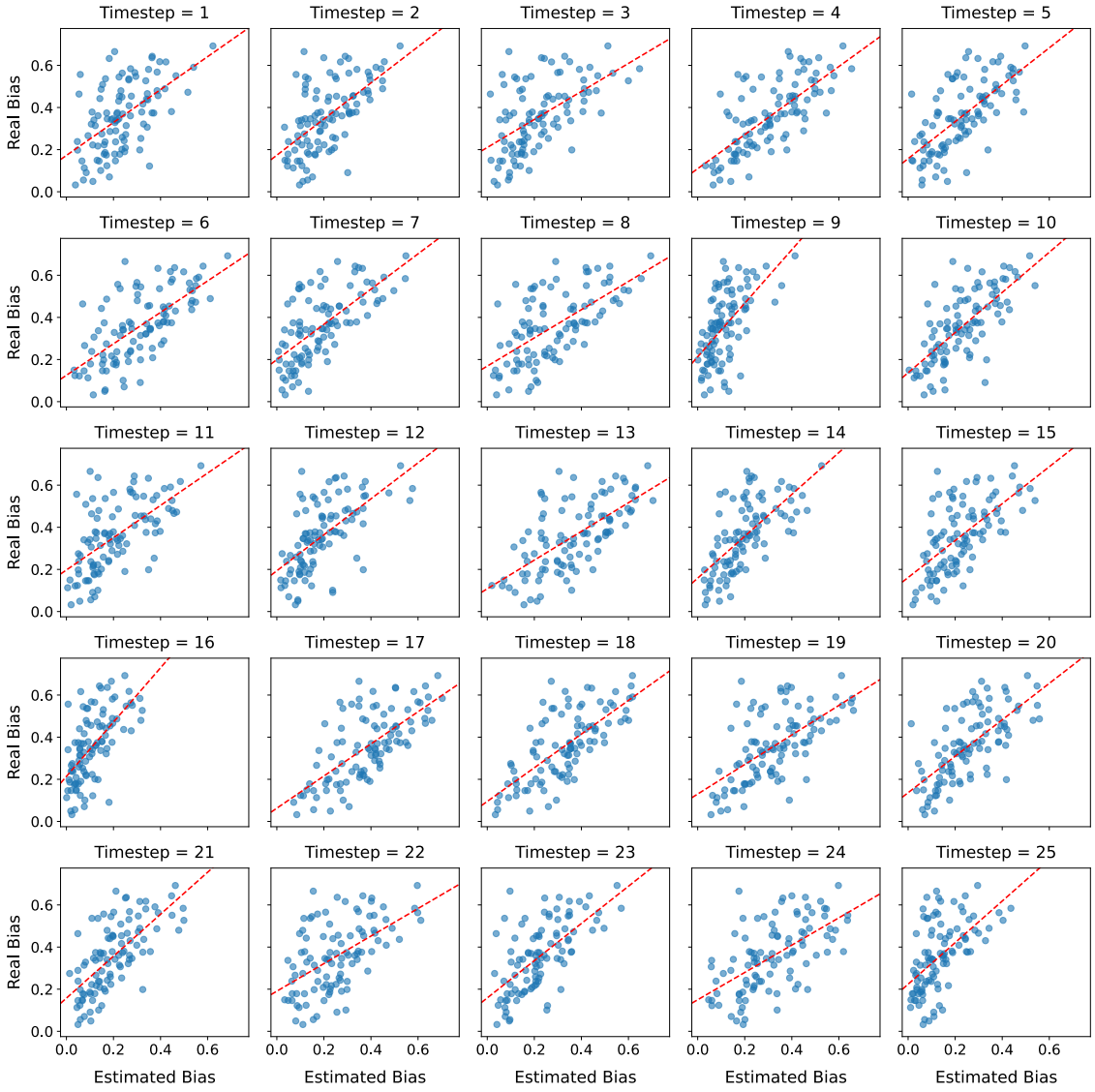


Figure 19: Comparison of real and estimated race biases for 100 occupations (blue dots) as well as their correlation trend line in the G.x.R. debiasing setup (timesteps 25 to 1).

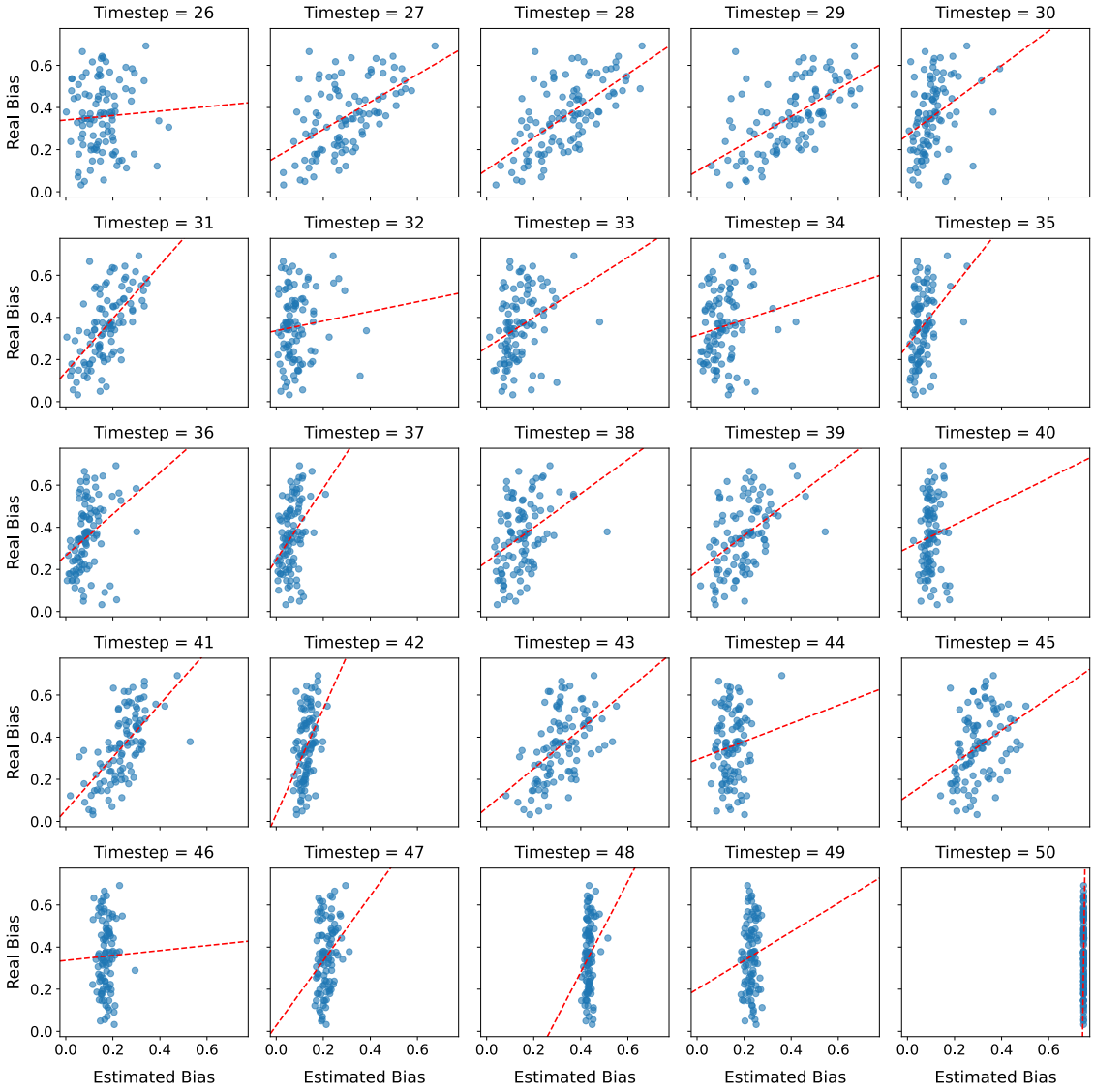


Figure 20: Comparison of real and estimated race biases for 100 occupations (blue dots) as well as their correlation trend line in the G.x.R. debiasing setup (timesteps 50 to 26).

References

Xiang An, Jiangkang Deng, Jia Guo, Ziyong Feng, Xuhan Zhu, Yang Jing, and Liu Tongliang (2022). "Killing Two Birds with One Stone: Efficient and Robust Training of Face Recognition CNNs by Partial FC". In: *CVPR*.

Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli (2022). "data2vec: A General Framework for Self-supervised Learning in Speech, Vision and Language." In: *ICML*. Ed. by Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato. Vol. 162. Proceedings of Machine Learning Research. PMLR, pp. 1298–1312.

Hritik Bansal, Da Yin, Masoud Monajatipoor, and Kai-Wei Chang (2022). "How well can Text-to-Image Generative Models understand Ethical Natural Language Interventions?" In: *EMNLP*. Ed. by Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang. Association for Computational Linguistics, pp. 1358–1370.

Federico Bianchi, Pratyusha Kalluri, Esin Durmus, Faisal Ladhak, Myra Cheng, Debora Nozza, Tatsunori Hashimoto, Dan Jurafsky, James Zou, and Aylin Caliskan (2023). "Easily accessible text-to-image generation amplifies demographic stereotypes at large scale". In: *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, pp. 1493–1504.

Steven Bird, Ewan Klein, and Edward Loper (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Abeba Birhane, Vinay Uday Prabhu, and Emmanuel Kahembwe (2021). "Multimodal datasets: misogyny, pornography, and malignant stereotypes". In: *arXiv*. cite arxiv:2110.01963Comment: 33 pages.

BLS (2024). *Employment Situation - 2024 M08 Results — bls.gov*. https://www.bls.gov/news.release/empsit.toc.htm. [Accessed 03-10-2024].

Manuel Brack, Felix Friedrich, Dominik Hintersdorf, Lukas Struppek, Patrick Schramowski, and Kristian Kersting (2023). "Sega: Instructing text-to-image models using semantic guidance". In: *Advances in Neural Information Processing Systems* 36, pp. 25365–25389.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin (2021). "Emerging Properties in Self-Supervised Vision Transformers." In: *ICCV*. IEEE, pp. 9630–9640.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon (2022). "Perception Prioritized Training of Diffusion Models." In: *CVPR*. IEEE, pp. 11462–11471.

Kristy Choi, Aditya Grover, Trisha Singh, Rui Shu, and Stefano Ermon (2020). "Fair Generative Modeling via Weak Supervision." In: *ICML*. Vol. 119. Proceedings of Machine Learning Research. PMLR, pp. 1887–1898.

Yujin Choi, Jinseong Park, Hoki Kim, Jaewook Lee, and Saerom Park (2024). "Fair Sampling in Diffusion Models through Switching Mechanism." In: *AAAI*. Ed. by Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan. AAAI Press, pp. 21995–22003.

Min Jin Chong and David A. Forsyth (2020). "Effectively Unbiased FID and Inception Score and Where to Find Them." In: *CVPR*. Computer Vision Foundation / IEEE, pp. 6069–6078.

Ching-Yao Chuang, Varun Jampani, Yuanzhen Li, Antonio Torralba, and Stefanie Jegelka (2023). "Debiasing Vision-Language Models via Biased Prompts." In: *CoRR* abs/2302.00070.

CommonCrawl (2024). *Common Crawl - Overview — commoncrawl.org*. https://commoncrawl.org/overview. [Accessed 30-09-2024].

Corinna Cortes and Vladimir Vapnik (1995). "Support-vector networks". In: *Machine learning* 20.3, pp. 273–297.

Zoe De Simone, Angie Boggust, Arvind Satyanarayan, and Ashia Wilson (2023). "What is a Fair Diffusion Model? Designing Generative Text-To-Image Models to Incorporate Various Worldviews". In: *arXiv preprint arXiv:2309.09944*.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, pp. 4171–4186.

Prafulla Dhariwal and Alexander Quinn Nichol (2021a). "Diffusion models beat gans on image synthesis". In: *Advances in neural information processing systems* 34, pp. 8780–8794.

Prafulla Dhariwal and Alexander Quinn Nichol (18–24 Jul 2021b). "Improved Denoising Diffusion Probabilistic Models". In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 8162–8171.

Stefano Ermon (2023). *Stanford course CS236 - Fall 2023 - Lecture 9: Generative Adverserial Networks*. https://deepgenerativemodels.github.io/assets/slides/cs236_lecture9.pdf. [Accessed 20-09-2024].

Piero Esposito, Parmida Atighehchian, Anastasis Germanidis, and Deepti Ghadiyaram (2023). "Mitigating stereotypical biases in text to image generative systems". In: *arXiv preprint arXiv:2310.06904*.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach (2024). "Scaling Rectified Flow Transformers for High-Resolution Image Synthesis." In: *CoRR* abs/2403.03206.

Research Facebook (2024). *facebook/dinov2-large · Hugging Face — huggingface.co*. https://huggingface.co/facebook/dinov2-large. [Accessed 01-10-2024].

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning (Nov. 3, 2006). "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling." In: *Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*. Ed. by The Association for Computer Linguistics, pp. 363–370.

Ronald Aylmer Fisher (1935). "The detection of linkage with "dominant" abnormalities". In: *Annals of Eugenics* 6.2, pp. 187–201.

Kathleen C. Fraser, Isar Nejadgholi, and Svetlana Kiritchenko (2023). "A Friendly Face: Do Text-to-Image Systems Rely on Stereotypes when the Input is Under-Specified?" In: *The AAAI-23 Workshop on Creative AI Across Modalities*.

Felix Friedrich, Patrick Schramowski, Manuel Brack, Lukas Struppek, Dominik Hintersdorf, Sasha Luccioni, and Kristian Kersting (2023). "Fair Diffusion: Instructing Text-to-Image Generation Models on Fairness." In: *CoRR* abs/2302.10893.

Rohit Gandikota, Hadas Orgad, Yonatan Belinkov, Joanna Materzynska, and David Bau (2024). "Unified Concept Editing in Diffusion Models." In: *WACV*. IEEE, pp. 5099–5108.

Sara Ghazanfari, Alexandre Araujo, Prashanth Krishnamurthy, Farshad Khorrami, and Siddharth Garg (2024). "LipSim: A Provably Robust Perceptual Similarity Metric". In: *The Twelfth International Conference on Learning Representations*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Y. Bengio (June 2014). "Generative Adversarial Networks". In: *Advances in Neural Information Processing Systems* 3.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778.

Jeff Heaton (2017). "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning". In: *Genetic Programming and Evolvable Machines* 19, pp. 305–307.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi (2021). "CLIPScore: A Reference-free Evaluation Metric for Image Captioning." In: *EMNLP (1)*. Ed. by Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen tau Yih. Association for Computational Linguistics, pp. 7514–7528.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter (2017). "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium." In: *NIPS*. Ed. by Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, pp. 6626–6637.

Jonathan Ho, Ajay Jain, and Pieter Abbeel (2020). "Denoising Diffusion Probabilistic Models." In: *NeurIPS*. Ed. by Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin.

Jonathan Ho and Tim Salimans (2021). "Classifier-Free Diffusion Guidance". In: *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.

Hyejin Hong, Sohyeon Kim, and Jee Hang Lee (2022). "On the Evaluation of Generated Stylised Lyrics Using Deep Generative Models: A Preliminary Study." In: *ICHI*. Ed. by Hakimjon Zaynidinov, Madhusudan Singh, Uma Shanker Tiwary, and Dhananjay Singh. Vol. 13741. Lecture Notes in Computer Science. Springer, pp. 132–139.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd (2020). "spaCy: Industrial-strength Natural Language Processing in Python". In.

Andrew Howard, Ruoming Pang, Hartwig Adam, Quoc V. Le, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, Mingxing Tan, Grace Chu, Vijay Vasudevan, and Yukun Zhu (2019). "Searching for MobileNetV3." In: *ICCV*. IEEE, pp. 1314–1324.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen (2022). "LoRA: Low-Rank Adaptation of Large Language Models." In: *ICLR*. OpenReview.net.

Aapo Hyvärinen (2005). "Estimation of Non-Normalized Statistical Models by Score Matching." In: *J. Mach. Learn. Res.* 6, pp. 695–709.

A. Imran and Demetri Terzopoulos (2019). "Multi-adversarial Variational Autoencoder Networks". In: *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, pp. 777–782.

Christopher Jarzynski (1997). "Equilibrium free-energy differences from nonequilibrium measurements: A master-equation approach". In: *Physical Review E* 56.5, p. 5018.

Marco Jiralerspong, Avishek Joey Bose, and Gauthier Gidel (2023). "Feature Likelihood Score: Evaluating Generalization of Generative Models Using Samples." In: *CoRR* abs/2302.04440.

Kimmo Kärkkäinen and Jungseock Joo (2021). "FairFace: Face Attribute Dataset for Balanced Race, Gender, and Age for Bias Measurement and Mitigation." In: *WACV*. IEEE, pp. 1547–1557.

Sergey Kastryulin, Jamil Zakirov, Denis Prokopenko, and Dmitry V. Dylov (2022). *PyTorch Image Quality: Metrics for Image Quality Assessment*.

Eunji Kim, Siwon Kim, Chaehun Shin, and Sungroh Yoon (2023). "De-stereotyping text-to-image models through prompt tuning.(2023)". In: *ICML 2023*.

Yeongmin Kim, Byeonghu Na, Minsang Park, JoonHo Jang, Dongjun Kim, Wanmo Kang, and Il-Chul Moon (2024). "Training Unbiased Diffusion Models From Biased Dataset." In: *ICLR*. OpenReview.net.

Younghyun Kim, Sangwoo Mo, Minkyu Kim, Kyungmin Lee, Jaeho Lee, and Jinwoo Shin (2023). "Bias-to-text: Debiasing unknown visual biases through language interpretation". In: *arXiv preprint arXiv:2301.11104*.

D. Kingma (2017). *Variational Inference and Deep Learning: A New Synthesis*. Ridderprint.

Diederik Kingma and Ruiqi Gao (2024). "Understanding diffusion objectives as the elbo with simple data augmentation". In: *Advances in Neural Information Processing Systems* 36.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho (2021). "Variational diffusion models". In: *Advances in neural information processing systems* 34, pp. 21696–21707.

Diederik P Kingma and Ruiqi Gao (2023). "Understanding Diffusion Objectives as the ELBO with Simple Data Augmentation". In: *Thirty-seventh Conference on Neural Information Processing Systems*.

Diederik P. Kingma and Jimmy Ba (2014). *Adam: A Method for Stochastic Optimization*. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015.

Diederik P. Kingma and Ruiqi Gao (2024). "Understanding diffusion objectives as the ELBO with simple data augmentation". In: *Proceedings of the 37th International Conference on Neural Information Processing Systems*. NIPS '23. Curran Associates Inc.

Diederik P. Kingma and Max Welling (2014a). "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*. arXiv: http://arxiv.org/abs/1312.6114v10 [stat.ML].

Diederik P. Kingma and Max Welling (2014b). "Auto-Encoding Variational Bayes." In: *ICLR*. Ed. by Yoshua Bengio and Yann LeCun.

Mingi Kwon, Jaeseok Jeong, and Youngjung Uh (2023). "Diffusion Models Already Have A Semantic Latent Space". In: *The Eleventh International Conference on Learning Representations*.

Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen (2023). "The Role of ImageNet Classes in Fréchet Inception Distance." In: *ICLR*. OpenReview.net.

Jia Li, Lijie Hu, Jingfeng Zhang, Tianhang Zheng, Hua Zhang, and Di Wang (2024). *Fair Text-to-Image Diffusion via Fair Mapping*.

Ruizhe Li, Xiao Li, Guanyi Chen, and Chenghua Lin (Dec. 2020). "Improving Variational Autoencoder for Text Modelling with Timestep-Wise Regularisation". In: *Proceedings of the 28th International Conference on Computational Linguistics*. Ed. by Donia Scott, Nuria Bel, and Chengqing Zong. International Committee on Computational Linguistics, pp. 2381–2397.

Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao (2022). "Pseudo Numerical Methods for Diffusion Models on Manifolds." In: *ICLR*. OpenReview.net.

Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang (2015). "Deep Learning Face Attributes in the Wild." In: *ICCV*. IEEE Computer Society, pp. 3730–3738.

Alexandra Luccioni and Joseph Viviano (Aug. 2021). "What's in the Box? An Analysis of Undesirable Content in the Common Crawl Corpus". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Ed. by Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli. Association for Computational Linguistics, pp. 182–189.

Sasha Luccioni, Christopher Akiki, Margaret Mitchell, and Yacine Jernite (2024). "Stable bias: Evaluating societal representations in diffusion models". In: *Advances in Neural Information Processing Systems* 36.

Calvin Luo (2022). "Understanding Diffusion Models: A Unified Perspective." In: *CoRR* abs/2208.11970.

Vongani H Maluleke, Neerja Thakkar, Tim Brooks, Ethan Weber, Trevor Darrell, Alexei A Efros, Angjoo Kanazawa, and Devin Guillory (2022). "Studying bias in gans through the lens of race". In: *European Conference on Computer Vision*. Springer, pp. 344–360.

Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan (2021). "A survey on bias and fairness in machine learning". In: *ACM computing surveys (CSUR)* 54.6, pp. 1–35.

Andriy Mnih and Karol Gregor (2014). "Neural Variational Inference and Learning in Belief Networks." In: *ICML*. Vol. 32. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 1791–1799.

Gaspard Monge (1781). "Mémoire sur la théorie des déblais et des remblais". In: *Mem. Math. Phys. Acad. Royale Sci.*, pp. 666–704.

Stanislav Morozov, Andrey Voynov, and Artem Babenko (2021). "On Self-Supervised Image Representations for GAN Evaluation." In: *ICLR*. OpenReview.net.

Eliya Nachmani, Robin San-Roman, and Lior Wolf (2021). "Non Gaussian Denoising Diffusion Models." In: *CoRR* abs/2106.07582.

Ranjita Naik and Besmira Nushi (2023). "Social biases through the text-to-image generation lens". In: *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 786–808.

Charlie Nash, Jacob Menick, Sander Dieleman, and Peter W. Battaglia (2021). "Generating images with sparse representations." In: *ICML*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 7958–7968.

Radford M. Neal (1998). "Annealed importance sampling". In: *Statistics and Computing* 11, pp. 125–139.

Carolina Lopez Olmos, Alexandros Neophytou, Sunando Sengupta, and Dim P. Papadopoulos (2024). "Latent Directions: A Simple Pathway to Bias Mitigation in Generative AI." In: *CoRR* abs/2406.06352.

OpenAI (2024). *openai/clip-vit-large-patch14 · Hugging Face — huggingface.co*. https://huggingface.co/openai/clip-vit-large-patch14. [Accessed 01-10-2024].

Hadas Orgad, Bahjat Kawar, and Yonatan Belinkov (2023). "Editing Implicit Assumptions in Text-to-Image Diffusion Models." In: *ICCV*. IEEE, pp. 7030–7038.

Rishubh Parihar, Abhijnya Bhat, Abhipsa Basu, Saswat Mallick, Jogendra Nath Kundu, and R Venkatesh Babu (2024). "Balancing Act: Distribution-Guided Debiasing in Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6668–6678.

Gaurav Parmar, Richard Zhang, and Jun-Yan Zhu (2022). "On Aliased Resizing and Surprising Subtleties in GAN Evaluation." In: *CVPR*. IEEE, pp. 11400–11410.

William Peebles and Saining Xie (2023). "Scalable Diffusion Models with Transformers." In: *ICCV*. IEEE, pp. 4172–4182.

Malsha V Perera and Vishal M Patel (2023). "Analyzing bias in diffusion-based face generation models". In: *2023 IEEE International Joint Conference on Biometrics (IJCB)*. IEEE, pp. 1–10.

Patrick von Platen, Suraj Patil, Anton Lozhkov, Pedro Cuenca, Nathan Lambert, Kashif Rasul, Mishig Davaadorj, Dhruv Nair, Sayak Paul, William Berman, Yiyi Xu, Steven Liu, and Thomas Wolf (2022). *Diffusers: State-of-the-art diffusion models*. https://github.com/huggingface/diffusers.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach (2024). "SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis". In: *The Twelfth International Conference on Learning Representations*.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever (2021). "Learning Transferable Visual Models From Natural Language Supervision." In: *ICML*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, pp. 8748–8763.

Royi Rassin, Eran Hirsch, Daniel Glickman, Shauli Ravfogel, Yoav Goldberg, and Gal Chechik (2024). "Linguistic binding in diffusion models: Enhancing attribute correspondence through attention map alignment". In: *Advances in Neural Information Processing Systems* 36.

Suman Ravuri and Oriol Vinyals (2019). "Seeing is not necessarily believing: Limitations of biggans for data augmentation". In.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra (2014). "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *ICML (2014)*. cite arxiv:1401.4082Comment: Appears In Proceedings of the 31st International Conference on Machine Learning (ICML), JMLR: W&CP volume 32, 2014.

Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer (2021). "High-Resolution Image Synthesis with Latent Diffusion Models". In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. Springer International Publishing, pp. 234–241.

runwayml (2024). *runwayml/stable-diffusion-v1-5 · Hugging Face — huggingface.co*. https://huggingface.co/runwayml/stable-diffusion-v1-5. [Accessed 29-08-2024].

Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen (2016). "Improved Techniques for Training GANs." In: *NIPS*. Ed. by Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, pp. 2226–2234.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. (2022). "Laion-5b: An open large-scale dataset for training next generation image-text models". In: *Advances in Neural Information Processing Systems* 35, pp. 25278–25294.

Xudong Shen, Chao Du, Tianyu Pang, Min Lin, Yongkang Wong, and Mohan S. Kankanhalli (2024). "Finetuning Text-to-Image Diffusion Models for Fairness." In: *ICLR*. OpenReview.net.

Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou (2019). "Interpreting the Latent Space of GANs for Semantic Face Editing". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9240–9249.

Karen Simonyan and Andrew Zisserman (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *International Conference on Learning Representations*.

Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli (2015). "Deep Unsupervised Learning using Nonequilibrium Thermodynamics." In: *ICML*. Ed. by Francis R. Bach and David M. Blei. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, pp. 2256–2265.

Jiaming Song, Chenlin Meng, and Stefano Ermon (2021). "Denoising Diffusion Implicit Models." In: *ICLR*. OpenReview.net.

Yang Song and Stefano Ermon (2019). "Generative Modeling by Estimating Gradients of the Data Distribution." In: *NeurIPS*. Ed. by Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché Buc, Emily B. Fox, and Roman Garnett, pp. 11895–11907.

Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole (2021). "Score-Based Generative Modeling through Stochastic Differential Equations." In: *ICLR*. OpenReview.net.

StabilityAI (2024a). *Join the Stable Diffusion Discord Server! — discord.com*. https://discord.com/invite/stablediffusion. [Accessed 05-09-2024].

StabilityAI (2024b). *Terms of Use — Stability AI — stability.ai*. https://stability.ai/terms-of-use. [Accessed 06-09-2024].

George Stein, Jesse C. Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Leigh Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L. Caterini, Eric Taylor, and Gabriel Loaiza-Ganem (2023). "Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models". In: *Thirty-seventh Conference on Neural Information Processing Systems*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna (2016). "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.

Christopher T. H. Teo, Milad Abdollahzadeh, and Ngai-Man Cheung (2023). "Fair Generative Models via Transfer Learning." In: *AAAI*. Ed. by Brian Williams, Yiling Chen, and Jennifer Neville. AAAI Press, pp. 2429–2437.

Lucas Theis, Aäron van den Oord, and Matthias Bethge (2016). "A note on the evaluation of generative models." In: *ICLR*. Ed. by Yoshua Bengio and Yann LeCun.

Michalis Titsias and Miguel Lázaro-Gredilla (22–24 Jun 2014). "Doubly Stochastic Variational Bayes for non-Conjugate Inference". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. PMLR, pp. 1971–1979.

C. Tu and Yi-Fu Chen (2019). "Facial Image Inpainting with Variational Autoencoder". In: *2019 2nd International Conference of Intelligent Robotic and Control Engineering (IRCE)*, pp. 119–122.

Soobin Um, Suhyeon Lee, and Jong Chul Ye (2024). "Don't Play Favorites: Minority Guidance for Diffusion Models". In: *The Twelfth International Conference on Learning Representations*.

Arash Vahdat, Karsten Kreis, and Jan Kautz (2021). "Score-based Generative Modeling in Latent Space". In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., pp. 5998–6008.

Pascal Vincent (2011). "A Connection Between Score Matching and Denoising Autoencoders." In: *Neural Comput.* 23.7, pp. 1661–1674.

Tao Wang, Junzhe Liu, Cong Jin, Jianguang Li, and Shihao Ma (2020). "An intelligent music generation based on Variational Autoencoder". In: *2020 International Conference on Culture-oriented Science Technology (ICCST)*, pp. 394–398.

Zihao Wang, Lin Gui, Jeffrey Negrea, and Victor Veitch (2023). "Concept Algebra for (Score-Based) Text-Controlled Generative Models". In: *Thirty-seventh Conference on Neural Information Processing Systems*.

Zijie J. Wang, Evan Montoya, David Munechika, Haoyang Yang, Benjamin Hoover, and Duen Horng Chau (2023). "DiffusionDB: A Large-scale Prompt Gallery Dataset for Text-to-Image Generative Models." In: *ACL (1)*. Ed. by Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki. Association for Computational Linguistics, pp. 893–911.

Max Welling and Yee Whye Teh (2011). "Bayesian Learning via Stochastic Gradient Langevin Dynamics." In: *ICML*. Ed. by Lise Getoor and Tobias Scheffer. Omnipress, pp. 681–688.

Robert Wolfe and Aylin Caliskan (2022). "Markedness in Visual Semantic AI." In: *FAccT*. ACM, pp. 1269–1279.

Worldbank (2024). *World Bank Open Data website*. WorldBankOpenData . (n . d .) . WorldBankOpenData . https://data.worldbank.org/indicator/IT.NET.USER.ZS. [Accessed 08-08-2024].

Depeng Xu, Shuhan Yuan, Lu Zhang, and Xintao Wu (2018). "FairGAN: Fairness-aware Generative Adversarial Networks." In: *IEEE BigData*. Ed. by Naoki Abe, Huan Liu, Calton Pu, Xiaohua Hu, Nesreen K. Ahmed, Mu Qiao, Yang Song, Donald Kossmann, Bing Liu, Kisung Lee, Jiliang Tang, Jingrui He, and Jeffrey S. Saltz. IEEE, pp. 570–575.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park (2024). "One-step diffusion with distribution matching distillation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang (2018). *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. cite arxiv:1801.03924Comment: Accepted to CVPR 2018; Code and data available at https://www.github.com/richzhang/PerceptualSimilarity.

List of Figures

List of Tables