Taming Normalizing Flows

Shimon Malnick
Tel Aviv University
malnick@mail.tau.ac.il

Shai Avidan
Tel Aviv University
avidan@eng.tau.ac.il

Ohad Fried
Reichman University
ofried@runi.ac.il

Project page: https://shimonmalnick.github.io/taming_norm_flows

Abstract

We propose an algorithm for taming Normalizing Flow models — changing the probability that the model will produce a specific image or image category. We focus on Normalizing Flows because they can calculate the exact generation probability likelihood for a given image. We demonstrate taming using models that generate human faces, a subdomain with many interesting privacy and bias considerations. Our method can be used in the context of privacy, e.g., removing a specific person from the output of a model, and also in the context of debiasing by forcing a model to output specific image categories according to a given distribution. Taming is achieved with a fast fine-tuning process without retraining from scratch, achieving the goal in a matter of minutes. We evaluate our method qualitatively and quantitatively, showing that the generation quality remains intact, while the desired changes are applied.

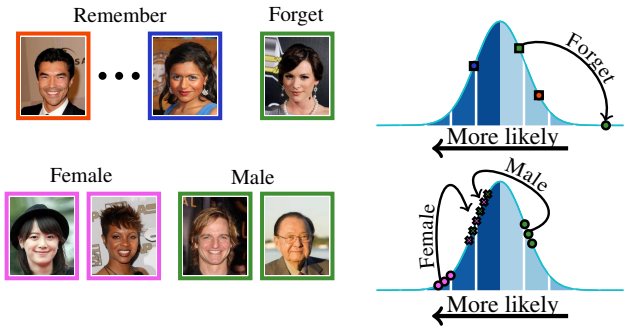

Figure 1. **Applications of our method.** A demonstration of our method for two different purposes, by changing the generation likelihood of different images. Colors of image frames correspond to colors of graph points. (**Top**) A model was tamed to reduce the likelihood of an image (Forget), while preserving the likelihood for the rest of the distribution (Remember). (**Bottom**) Debiasing a model that generates female faces with higher probability than males[1], so as to balance the generation likelihood.

1. Introduction

Generative models are increasing in popularity [26], partly due to the exponential growth in deep neural network techniques [19, 29, 35, 43]. In this work, we focus on generative models of human faces which, some might say, are becoming dangerously powerful. Synthetic images or videos of real people can be easily generated and used to spread misinformation [60], to harass [46], and to con [14]. Thus, a company developing a generative model might be interested, before releasing it to the public, in preventing the model from synthesizing the faces of certain celebrities[2]. Furthermore, a generative model might have been trained on biased data and thus under-represent certain groups of the population. In this case, it would be desirable to debias the model[3]. In the same spirit, generative models are often more likely to synthesize images of the individuals that were used for training the model [8, 9]. But, following the GDPR [48] "Right to Be Forgotten" approach, a person may request the company to re-train the model without their images — a just cause, but also a time consuming and expensive process.

Common to all these scenarios is the need to change the probability of a *given* pretrained generative model to generate certain individuals or demographics. In short, what is needed is a method to *tame* a model. In this work, we take a step towards solving this problem, suggesting an algorithm to tame *normalizing flows* [15, 16, 34, 37, 49]. We focus on normalizing flows because they provide an explicit probabilistic density function along a bijection between the image and latent spaces. We tame the model by fine-tuning it while constraining its output distribution. The constraint is twofold: forcing resemblance to the original model's dis-

[1] We trained a model on CelebA [41], containing 15% more females than males with a binary Male/Female label. We hope that future datasets will annotate gender more fluidly.

[2] For example: https://labs.openai.com/policies/content-policy

[3] For example: https://openai.com/blog/reducing-bias-and-improving-safety-in-dall-e-2/

tribution, while also adhering to the target probability of the taming process. We refer to these two different aspects of taming as *remembering* and *forgetting*, describing whether we wish to preserve the model's behavior (remember) or guide the model away from some outputs (forget)[4].

We propose a novel task: controlling probability estimating generative models such as Normalizing Flows, with a fast and simple approach, taming the model in minutes. We demonstrate our method on human faces — a domain with inherent privacy and bias concerns that our method can alleviate. After taming, the output distribution changes according to the target distribution, while preserving image generation quality. Our method applies probability measures directly to our loss and evaluation process. An illustration of different applications of our method is shown in Fig. 1.

Our main contributions are: (1) a general technique for taming normalizing flows, (2) applications of taming for privacy, by censoring specific individuals from the model's output, and (3) applications of taming for fairness, by modifying the frequency of attributes (*e.g.*, male vs. female) in the model's output.

2. Related work

Density estimation. Many generative models use *maximum likelihood* [40] to provide parametric density estimations. Some approaches are implicit such as GANs [19], while others are explicit, such as VAEs [35] and the recent state-of-the-art Diffusion Models [29, 53]. We focus on explicit approaches, specifically on models that provide an explicit tractable probability density function [49, 56, 57], since we can use that density to evaluate and quantify whether we move images away from the density modes.

Model editing. Generative model editing deals with methods that fine-tune a model in order to apply small changes to it. Bau *et al*. [5] allow users to choose specific changes on generated images and fine-tune the model's weights to apply them. Wang *et al*. [58] apply a user-chosen image warp on several examples to later fine-tune a model that produces images according to the warp. Cherepkov *et al*. [12] fine-tune a model to incorporate semantic changes and discover emerging semantics, but cannot edit a model according to a pre-determined goal.

Our work differs from the aforementioned methods that alter the behavior of the model globally. We, on the other hand, provide the ability to focus on specific areas in the latent space, without changing the whole domain. For a multi-modal generative model this virtue is vital, as local changes can be relevant only to specific outputs that reside in a specific mode. For a face generating model, instead of changing an attribute across all outputs, *e.g*. forcing a smile, our method enables elimination only of specific images of people that do not smile. Moreover, since our method uses a normalizing flow, in contrast to the methods above that use a GAN [19], we provide an exact evaluation of the latent distribution edit that we perform.

Debiasing models. Deep learning models are now integrated in many crucial systems, *e.g*., finance [3] and medical diagnosis [1]. Thus, ensuring the fairness of these models is crucial. There are various approaches to reduce model bias. Pre-processing and in-processing approaches, *e.g*., changing the training data [6, 17, 47] and using different training loss modulation techniques [4, 59, 63]. Unlike these methods, our work can be used on a *given* pre-trained model, without any prior demand on the training data.

Some post-processing methods constrain the sampling space [13, 30, 54, 61] but assume low-dimensional latent spaces (such as in GANs [19] and VAEs [35]), making them not suitable for normalizing flows. Restricting certain queries is also problematic as it allows easy deception [44].

Our approach changes the model itself, instead of changing the sampling space. This allows us to control models, as they are integrated in constantly changing environments.

Closest to our work is Kong and Chaudhuri [38] that proposed a method that enables data forgetting from a pre-trained GAN, which can also be used for debiasing. In contrast, our work is demonstrated on normalizing flows, providing an exact probability density evaluation of the edits. Moreover, our method can be applied locally on much less data, as shown in Sec. 4.1.

Continual learning. Continual learning (also known as lifelong-learning) is the field of teaching new tasks to a model sequentially. A fundamental problem in this domain, described as *catastrophic forgetting* [20, 36], is that while learning new tasks, models tend to forget the previous ones. We discuss how adjusting a normalizing flow can alter the generation probability of specific data. This can be thought of as teaching the model a new task (reducing the probability of some outputs), while preserving the knowledge of the original task (generating images as the model did before), similarly to continual learning. While relevant work in this field focuses on preventing forgetting, we can also choose to forget. In addition, prior work has focused mainly on discriminative tasks rather than generative ones.

Machine unlearning. Machine unlearning [7] refers to the process of removing the effect that certain training data have on a model's weights after training [11, 23]. If a user requests to delete their data, some privacy regulations [48] require the data to be deleted, together with the effect it had on any models trained on it. As opposed to approaches that aim to delete training data from trained models [18, 55, 62], we focus on changing the model's behavior regardless of whether the data we deal with belongs to the training set or not. Carlini *et al*. [8, 9] and Haim *et al*. [24] demon-

[4] Our approach does not technically fit the terms of forgetting and remembering, but rather emphasizing or preserving vs. de-emphasizing or abandoning. For simplicity, we use the terms remember and forget.

strate methods that extract training data from models. In our work, we do not focus on leaking a model's training data, but rather on changing the model's behavior with respect to some data distribution.

3. Method

We first define the problem at hand, followed by some technical background and a description of our approach.

3.1. Problem definition

Taming a normalizing flow model involves modifying its behavior with respect to some data. This includes images of an identity it was trained on, training images sharing some property, or even images out of the training set. For consistency of exposition, we describe the taming procedure as decreasing likelihood of certain data points (*i.e.*, images). That is, forgetting these points. Nevertheless, we also use taming in the opposite direction — increasing the likelihood of certain data for model debiasing (Sec. 4.2).

After taming, the behavior of the model should remain the same across the entire output space, except for the specific data we choose to forget. This implies three conflicting important goals: (1) The probability of producing images from the set we are trying to forget should be close to zero. We refer to this goal as *forgetting*. (2) For all images except the ones we would like to forget, the **probability distribution** to produce these outputs should be as similar as possible to the original model. We refer to this goal as *remembering*. (3) The quality of image generation should stay intact. An illustration of this concept on a 2D toy example is shown in Fig. 2.

There are many types of generative models that can produce photorealistic faces [32–34]. In this work we focus on normalizing flows, as they explicitly represent the image distribution (unlike the implicit nature of, *e.g.*, GANs), meaning that we can reason about probabilities and incorporate them into our losses, as explained below.

3.2. Normalizing flows

A normalizing flow is an invertible transformation of a probability density from a simple distribution to a more complex one. The initial density "flows through", to yield a different, yet normalized, density, and thus it is called a normalizing flow. As shown in previous work [15, 16, 21, 50] and in more recent progress [10, 22, 28, 37], the key behind these models is training an invertible function that maps samples from the data distribution domain to a tractable and easily sampled latent domain. At inference, since the mapping is invertible, a mapping in the opposite direction allows the transition from latent vectors to the image space. We intend to model a parametric probability density function given a set of examples.

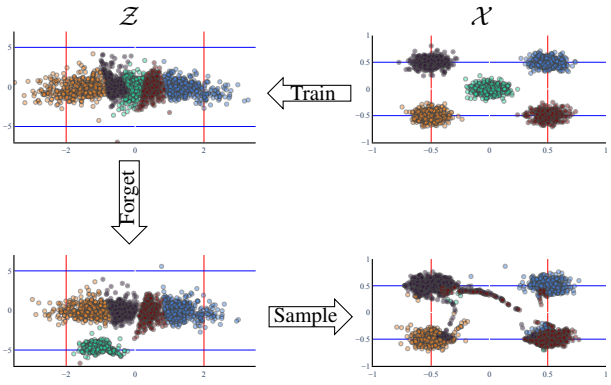

Figure 2. **A 2D example with a RealNVP [16] normalizing flow.** The image space $\mathcal{X}$ contains data points sampled from 5 Gaussians with different means. The prior distribution in the latent space $\mathcal{Z}$ is Gaussian. Given a normalizing flow that maps $\mathcal{Z} \to \mathcal{X}$, we tame the model to produce the same image space $\mathcal{X}$, apart from the middle Gaussian, which we wish to remove. (Each Gaussian represents a person, and we would like to forget one person.) (**Train**) Prior to our method, the inverse flow was trained to map points from $\mathcal{X}$ to $\mathcal{Z}$. To generate samples, the flow is used to map $\mathcal{Z}$ to $\mathcal{X}$. (**Forget**) We apply our method: latent vectors that were initially mapped to the center Gaussian now have a lower likelihood of being drawn from the prior distribution. (**Sample**) Now that we tamed the model, when we sample from $\mathcal{Z}$, the points are mapped (with high probability) to the 4 Gaussians.

Formally, let $Z \in \mathbb{R}^M$ be a random vector with a density function $\mathrm{p}_\theta(z)$ parameterized by $\theta \in \Theta$. Let $f_\theta : \mathbb{R}^M \to \mathbb{R}^M$ be a bijection function (parameterized by θ) with an inverse f_θ^{-1}, such that $f_\theta(Z) = X$ and $f_\theta^{-1}(X) = Z$. We denote the domain and range of f_θ as $\mathcal{Z}$ and $\mathcal{X}$ respectively, representing the latent and image spaces. For ease of notation, we also denote the density function of X as $\mathrm{p}_\theta(x)$. Using the formula of random variable change [15], we get:

$$\mathrm{p}_\theta(x) = \mathrm{p}_\theta(f_\theta^{-1}(x)) \cdot \left|\det\left(\mathbf{J}_{f_\theta^{-1}}(x)\right)\right|, \tag{1}$$

where $\mathbf{J}_{f_\theta^{-1}}(x) = \left[\frac{\partial f_\theta^{-1}(y)}{\partial y}\right]_{y=x}$ is the Jacobian matrix of f_θ^{-1} at x. Modern flows are built such that the determinant of the Jacobian is easily computed, usually by using a flow with a triangular Jacobian matrix. In these cases, using Eq. (1) we can construct a more tractable expression for the log-likelihood of the density $\mathrm{p}_\theta(x)$, using just the elements of the Jacobian's diagonal:

$$\log \mathrm{p}_\theta(x) = \log \mathrm{p}_\theta(f_\theta^{-1}(x)) + \sum_{i=1}^{M} \log \left|\mathbf{J}_{f_\theta^{-1}}(x)_{i,i}\right|. \tag{2}$$

We focus on modeling the latent space as a multivariate normal distribution with diagonal covariance, *i.e.*

$$\mathrm{p}_\theta(z) = \mathcal{N}\left(\mu_\theta, \sigma_\theta^2 \cdot I\right). \tag{3}$$

Since the covariance is diagonal, the prior is factorial, meaning we can easily decompose the density to univariate components:

$$\log \mathrm{p}_\theta(z) = \log \prod_{i=1}^{M} \mathrm{p}_\theta(z_i) = \sum_{i=1}^{M} \log \mathrm{p}_\theta(z_i), \tag{4}$$

where $\forall i \in \{1, \ldots, M\} : \mathrm{p}_\theta(z_i) \sim \mathcal{N}(\mu_{\theta i}, \sigma_{\theta i}^2)$.

Given an i.i.d. set of samples from the image distribution $\mathcal{D} = \{x_i\}_{i=1}^n \sim X$, we can use optimization methods [2] to estimate the parameters θ based on minimization of the average negative log-likelihood:

$$\mathrm{A}_\theta(\mathcal{D}) := -\frac{1}{n} \sum_{i=1}^{n} \log \mathrm{p}_\theta(x_i). \tag{5}$$

Given f_θ that was trained to minimize the term in Eq. (5), we assume that the Negative Log-Likelihood (NLL) of the model w.r.t. the image distribution is normal, *i.e.*:

$$\mathrm{N}_\theta(X) := -\log \mathrm{p}_\theta(X) \sim \mathcal{N}(\mu_\theta, \sigma_\theta^2). \tag{6}$$

We elaborate on this assumption in the next section and further in the Supplementary. This means that **the NLL of images in $\mathcal{D}$ is drawn from a normal distribution**, which will help us to use probabilistic measures that fit normal distributions.

3.3. Task

We wish to tame a pretrained *base* normalizing flow model f_{θ_B}, with parameters $\theta_B \in \Theta$ learned using a dataset $\mathcal{D}$. For taming we need a dataset $\mathcal{D}_R$ of images to be remembered, and a dataset $\mathcal{D}_F$ of images to be forgotten. The dataset $\mathcal{D}_R$ can be the one used to train the base model, or a different set of images representing a similar distribution, with a much smaller size. The result is a *tamed* model, with network weights θ_T, that adheres to the *remembering* and *forgetting* goals we introduced in Sec. 3.1.

Forgetting. We use the fact that normalizing flow models enable precise density evaluation, to set a forgetting threshold using the samples' likelihood. Since we have access to images we wish to remember, $\mathcal{D}_R$, we can estimate the likelihood of samples in it. To forget a set of images, we reduce their likelihood and compare it to the likelihood of images in $\mathcal{D}_R$.

To evaluate the success of forgetting, we need to define a proper threshold — how low should the likelihood be, for us to consider the sample forgotten? Naïvely choosing a hand-picked threshold for the likelihood is problematic — too small, and the forgetting process is unnecessarily hard, too large, and we may not forget enough. The problem is further complicated because working with likelihoods in the relatively high-dimensional latent space $\mathcal{Z}$ is not intuitive.

So instead of defining the threshold in absolute terms, we define it in relative terms. That is, an image is considered forgotten if its likelihood of being sampled is low compared to the distribution of the images to be remembered (Eq. (6)).

Switching to NLL for convenience, we assume that the NLL of images in $\mathcal{D}_R$ is normally distributed, and support this assumption with a *Kolmogorov-Smirnov test* [42]. The Supplementary contains additional information about the assumption and evaluation in case it does not hold. With this assumption, we can specify the forgetting threshold in units of standard deviation σ.

We denote the mean and standard deviation of the Normal distribution over the NLL values of images in $\mathcal{D}_R$ as:

$$\begin{aligned} \mu_R &= \mathop{\mathbb{E}}_{x \sim \mathcal{D}_R} \big[-\log \mathrm{p}_{\theta_T}(x) \big], \\ \sigma_R &= \sqrt{\mathop{\mathbb{E}}_{x \sim \mathcal{D}_R} \big[\big(-\log \mathrm{p}_{\theta_T}(x) - \mu_R \big)^2 \big]}. \end{aligned} \tag{7}$$

We define the threshold δ ($\delta = 4$ in our experiments), specified in standard deviation units, *i.e.* we wish that for an image $x \in \mathcal{D}_F$, its NLL will be far from μ_R by exactly $\delta \cdot \sigma_R$. We define a signed distance normalized in standard deviations (SD):

$$\mathrm{d}_{\mu_R, \sigma_R}(x, \delta; \theta_T) := \frac{-\log \mathrm{p}_{\theta_T}(x) - (\mu_R + \delta \cdot \sigma_R)}{\sigma_R}. \tag{8}$$

Observe that by specifying δ in terms of the NLL distribution's SD, we avoid the need to work directly with the actual NLL values in latent space $\mathcal{Z}$. We found that directly forgetting these examples by minimizing their likelihood (the opposite of Eq. (5)) was not stable, as the loss decreased by pushing some images with low likelihoods to even lower likelihoods while neglecting other images, which was harder to converge. Thus, we define our *forgetting* loss, that normalizes the distance between the images and the threshold:

$$\mathcal{L}_F(\theta_T, \mathcal{D}_F, \mathcal{D}_R) = \frac{1}{|\mathcal{D}_F|} \sum_{x \in \mathcal{D}_F} \mathrm{S}\big(\sigma_R^2 \, \mathrm{d}_{\mu_R, \sigma_R}^2(x, \delta; \theta_T)\big), \tag{9}$$

where $\mathrm{S}(\cdot)$ is the Sigmoid function [25]. Intuitively, this loss encourages every image in $\mathcal{D}_F$ to have a NLL that is as close to the threshold as possible.

Given an error parameter $\epsilon > 0$, the threshold is met when the likelihood of all examples in $\mathcal{D}_F$ is in a distance bounded by ϵ around the threshold:

$$\forall x \in \mathcal{D}_F : |\, \mathrm{d}_{\mu_R, \sigma_R}(x, \delta; \theta_T)| < \epsilon, \tag{10}$$

i.e., this is the stopping criteria for our method. ϵ controls the size of the error margin allowed around the threshold. When this criterion holds, the images in $\mathcal{D}_F$ have a very

low likelihood, compared to the images in $\mathcal{D}_R$ (see more in the Supplementary).

Remembering. We aim to remember the images in $\mathcal{D}_R$, *i.e.*, preserve the NLL distribution of the model w.r.t. these images. When we consider the entire distribution, we are not concerned with the NLL of each image seperatly, but rather the distribution as a whole. Assuming $\mathcal{D}_R$ is an i.i.d. set drawn from X, we compare the NLL distribution of the original and tamed models, *i.e.* we compare $\mathrm{N}_{\theta_B}(X)$ and $\mathrm{N}_{\theta_T}(X)$, respectively (see Eq. (6)). The closer these distributions are, the less impact our procedure had on the images that we did not intend to forget. We use the *KL divergence* [39] between these distributions to measure their proximity. We use both the forward and reverse KL divergence, denoted as $\mathcal{L}_{KL_F}(\theta_T, \theta_B; \mathcal{D}_R)$ and $\mathcal{L}_{KL_R}(\theta_T, \theta_B; \mathcal{D}_R)$ respectively, as explained in Sec. 5. Moreover, we also use the average NLL loss $\mathrm{A}_{\theta_T}(\mathcal{D}_R)$ (Eq. (5)), to maintain the original model's NLL on $\mathcal{D}_R$. Our combined *remembering* loss is thus:

$$\begin{aligned}\mathcal{L}_R(\theta_T, \theta_B, \mathcal{D}_R) &= (1-\gamma)\,\mathrm{A}_{\theta_T}(\mathcal{D}_R) \\ &+ \gamma\left(\mathcal{L}_{KL_F}\left(\theta_T, \theta_B; \mathcal{D}_R\right) + \mathcal{L}_{KL_R}\left(\theta_T, \theta_B; \mathcal{D}_R\right)\right),\end{aligned} \tag{11}$$

where γ controls the ratio between the original task and the explicit distribution proximity loss. As $\mathcal{D}_R$ can represent a distribution that is different than the original task (in case $\mathcal{D}_R$ is not the training set used to train θ_B), the loss acts in line with this distribution. The "closer" $\mathcal{D}_R$ is to the training set, the higher we preserve the image space of θ_B. Our total objective is a weighted combination of the forget and remember losses:

$$\begin{aligned}\theta_T = \arg\min_{\theta} \Big\{ &\alpha \cdot \mathcal{L}_F\left(\theta, \mathcal{D}_F, \mathcal{D}_R\right) + \\ &(1-\alpha)\cdot \mathcal{L}_R\left(\theta, \theta_B, \mathcal{D}_R\right)\Big\}.\end{aligned} \tag{12}$$

Algorithm 1 Normalizing Flow taming

Input: Normalizing Flow f_{θ_B}, Forget images $\mathcal{D}_F$, Remember images $\mathcal{D}_R$, Forget threshold $\delta > 0$, error bound $\epsilon > 0$

Hyperparameters: $\eta > 0, \alpha \in (0, 1)$

1: $\theta_T \leftarrow \theta_B$
2: **for** iteration $i = 1, 2, \ldots$ **do**
3: Sample batches $X_F \sim \mathcal{D}_F$, $X_R \sim \mathcal{D}_R$
4: Estimate distribution $(\mu_R, \sigma_R) \leftarrow -\log \mathrm{p}_{\theta_T}(X_R)$
5: $\vec{\mathrm{d}} = \mathrm{d}_{\mu_R, \sigma_R}(X_F, \delta; \theta_T)$
6: **if** $\forall i : |\,\mathrm{d}_i\,| < \epsilon$ **then**
7: **break**
8: $\mathcal{L} \leftarrow \alpha\mathcal{L}_F(\theta_T, X_F, X_R) + (1-\alpha)\mathcal{L}_R(\theta_T, \theta_B, X_R)$
9: $\theta_T \leftarrow \theta_T - \eta\nabla\mathcal{L}$
10: **Return** f_{θ_T}

We use SGD [2] to optimize the objective, stopping the process when Eq. (10) is satisfied. A summary of our method is presented in Algorithm 1.

4. Experiments

We conduct experiments that evaluate the generation probability reduction for images of a specific person, a set of people and people with specific attributes.

We use Glow [34] as our base model f_{θ_B}, trained on 128×128 images from the FFHQ [33] dataset and the CelebA [41] training set. The running time for our experiments is 3–40 minutes with no more than 800 iterations. Full technical details, can be found in the Supplementary. To improve our method's run-time, we compute the remember batch NLL distribution parameters (see Line 4 in Algorithm 1) every 10 iterations. In our analysis below, we focus on the effect on the forget set $\mathcal{D}_F$, as our method preserves the distribution on the remember set $\mathcal{D}_R$, as can be seen in the Supplementary.

4.1. Taming an identity

First, we examine the ability to reduce the generation probability of a person's images. This corresponds to many applications, *e.g.*, a model that violates a person's privacy can be censored this way.

In this experiment, we have access to $\mathcal{D}$, the training set used to train the given model. We wish to tame the model in such a way that images containing a specific identity will not be generated by the model, or at least to reduce this probability as we see fit. These images, denoted as $\mathcal{D}_F$, are a part of the training set, *i.e.* $\mathcal{D}_F \subset \mathcal{D}$. Therefore, the remember set in this case is defined as $\mathcal{D}_R := \mathcal{D} \setminus \mathcal{D}_F$. We run experiments with different sizes of $\mathcal{D}_F$, using the same training setting as the pre-trained model. Experiments are halted only when the stopping criteria is met (see Eq. (10)).

Forgetting evaluation. Let $\mathrm{F}^{(\mu,\sigma)}(x) := \Phi(\frac{x-\mu}{\sigma})$ be the CDF of normal R.V with parameters (μ, σ), where $\Phi(\cdot)$ is the CDF of the standard normal distribution. Then, an image with NLL x is forgotten if:

$$1 - \mathrm{F}^{(\mu,\sigma)}(x) \geq 1 - \mathrm{F}^{(\mu,\sigma)}(\mu + \delta\sigma) = 1 - \Phi(\delta), \tag{13}$$

which, for the case of $\delta = 4$, is approximately 3.2e−5. This means that if an image is forgotten, its NLL resides in the highest (worst) 0.0032% percentile of the NLL distribution that $\mathcal{D}_R$ is drawn from.

Measuring the success in forgetting an image with tamed model θ_T boils down to:

$$\mathrm{q}_{\theta_T}^{(\mu_R,\sigma_R)}(x) := 1 - \mathrm{F}^{(\mu_R,\sigma_R)}\left(-\log \mathrm{p}_{\theta_T}(x)\right), \tag{14}$$

which we denote as **Likelihood Quantile**, and generalized to a set of images using the mean. For brevity, we omit the distribution parameres and denote it as $\mathbf{q}_{\boldsymbol{\theta}}(\cdot)$.

Figure 3. **Visual generation evaluation.** These samples were generated using the base and tamed models. Can you tell which model was used for each image? If you cannot, it implies that our process preserves image quality. Answer (upside down): even columns — base, odd columns — tamed.

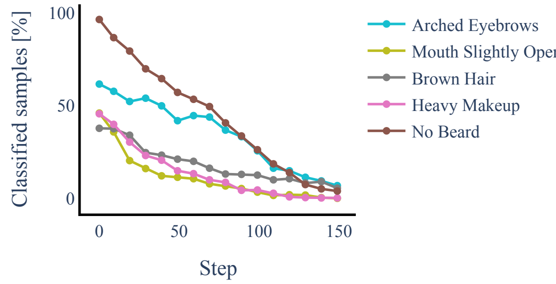

Figure 4. **Taming attributes.** The proportion of random samples that possess the attribute we aim to forget along our method. Each line represents an experiment to forget a specific attribute.

We are also interested in evaluating the "damage" inflicted to other image sets, *e.g.* showing that while forgetting $\mathcal{D}_F$ we did not harm the likelihood of images in $\mathcal{D}_R$. To do this, we examine how the likelihood quantile decreases between the base model and the tamed one, on some given set S. We denote this as the **Quantile Drop** $\mathbf{QD}_{\theta_B,\theta_T}(\cdot)$:

$$\mathrm{QD}_{\theta_B,\theta_T}(S) := \mathrm{q}_{\theta_B}(S) - \mathrm{q}_{\theta_T}(S). \tag{15}$$

We use this score to evaluate our method's effect on different sets of images:
- Holdout images of the forget identity — denoted $\mathcal{D}'_F$.
- Random images from $\mathcal{D}_R$.
- Holdout identities from the remember set — denoted $\mathcal{D}_R^{\text{id}}$.
- Closest identities (nearest neighbors in the face embedding space) in $\mathcal{D}_R$ — denoted as $\mathcal{D}_R^{\text{NN}}$.

Results are shown in Tab. 1, *e.g.*, the first row shows that we reach the threshold of Eq. (10) (meaning all images in the forget set are within the error bound range) while reducing the likelihood quantile of $\mathcal{D}_F$ by 0.47, and maintaining small likelihood changes for the rest of the examined sets. Additional results on a baseline and on a different modality can be found in the Supplementary.

While we do not force resemblance to the images we forget, our method is able to apply the local changes w.r.t. to the forgotten identity, while reducing the likelihood of unseen images of that identity ($\mathcal{D}'_F$) significantly. Moreover, comparing these likelihoods to the low likelihood quantiles of the nearest identities ($\mathcal{D}_R^{NN}$) suggests that our method can implicitly focus on the identity we forget, generalizing to unseen images, with minimal damage to identities that are "close" ($\mathcal{D}_R^{NN}$), and to other random identities ($\mathcal{D}_R^{id}$).

As seen in Fig. 3, our method does not have a significant impact on the quality of the generated images. Sec. 5 contains further analysis of the experiment. The Supplementary contains a more detailed analysis of this experiment, including evaluating the preservation of the $\mathcal{D}_R$ distribution, along with additional experiments using different thresholds and unseen forget identities as the forget set, *i.e.* $\mathcal{D}_F \not\subset \mathcal{D}$.

4.2. Taming an attribute

Next, we zoom out from the local effect on a small number of images, to a broader aspect of change. We show that our method can be applied even when the desired change is more general, *e.g.* reducing the probability of generated images that contain inappropriate content. Moreover, this experiment shows that we can apply our method on a big set of images, *i.e.* $|\mathcal{D}_F| \propto |\mathcal{D}|$.

In this experiment we have access to $\mathcal{D}$, the model's training set. We wish to forget an attribute common to many images in the training set, *e.g.* wearing glasses or smiling. Thus, we use a classifier for that property, denoted as $C : \mathcal{X} \to \{0, 1\}$, to define the remember and forget sets: $\mathcal{D}_F = \{x \in \mathcal{D} \mid C(x) = 1\}$, $\mathcal{D}_R = \mathcal{D} \setminus \mathcal{D}_F$.

As we wish to reduce the generation probability of images with some property, evaluation is straightforward, by passing random samples from the prior distribution through the normalizing flow, and classifying the output images. Our method alters the model, in order to reduce the number of outputs classified as possessing the relevant attribute. This approach can be used to debias a model, *e.g.* a model that generates images of a certain group of the population with high probability (to the point of over representation), one can use our method to reduce that probability. The results are shown in Fig. 4, showing we are able to tame a model to reduce the sampling rate of an attribute. This is applied on a large forget set $\mathcal{D}_F$, *e.g.*, forgetting the "No Beard" property holds: $|\mathcal{D}_F| = 135,779 = 0.83 \cdot |\mathcal{D}|$. The proportion of attribute changes can be controlled by the number of steps we run our process, as we can stop the process when the desired ratio of some property suffices.

Taming can also be used in the opposite direction, to increase the number of generated samples possessing the chosen attribute. This means that in this case, instead of forgetting a group of images, we do the opposite and increase the representation of these images in the output space.

Fig. 5 visualizes how we tame a model to generate less (or more) images with a chosen attribute (or attributes). While our goal is to control the generation probability of

# Images	Forget threshold	Quantile drop $\mathrm{QD}_{\theta_B,\theta_T}(\cdot)$ (see Eq. (15))				
		$\mathcal{D}_F(\uparrow)$	$\mathcal{D}'_F(\uparrow)$	$\mathcal{D}_R(\downarrow)$	$\mathcal{D}_R^{\mathrm{id}}(\downarrow)$	$\mathcal{D}_R^{\mathrm{NN}}(\downarrow)$
1	✓	0.47 ± 0.04	$< 10^{-2} \pm 0.03$	$< 10^{-2} \pm 0.01$	$< 10^{-3} \pm 0.01$	0.01 ± 0.01
4	✓	0.42 ± 0.17	0.05 ± 0.03	$< 10^{-2} \pm 0.00$	0.01 ± 0.01	0.03 ± 0.02
8	✓	0.34 ± 0.13	0.06 ± 0.04	$< 10^{-3} \pm 0.00$	$< 10^{-2} \pm 0.00$	0.01 ± 0.01
15	✓	0.38 ± 0.14	0.12 ± 0.04	$< 10^{-3} \pm 0.00$	$< 10^{-2} \pm 0.00$	0.01 ± 0.00

Table 1. **Forget identity effect.** When we forget images of the same identity, we are able to generalize and reduce the likelihood of unseen images of that identity ($\mathcal{D}'_F$), while maintaining the likelihood on the rest of the space (see Sec. 4.1 for more details). (↑) and (↓) are used to indicate whether higher or lower is better, respectively.

the distribution as a whole, we are able to preserve the identity in the generated images. This suggests that we preserve the structure of the given model's underlying latent space, and apply changes very specifically. Since there are many methods that utilize different properties of latent spaces in generative models [51, 52], this is useful for tasks that use an image as input, *e.g.* image editing and image translation. In the Supplementary we show more examples, model debiasing and discuss the effect of coupled attributes (*e.g.*, what happens if most eyeglass wearing people are male).

4.3. Taming without the training set

Next, we consider situations where we lack access to the model's training data. Instead, we assume access to different data from a similar distribution. An example of such a scenario can be a company that releases a generative model to the public, without the data on which it was trained. Entities using this model might want to alter the model w.r.t. different data, while maintaining the model's performance.

In this experiment the setup is similar to Sec. 4.1, except that $\mathcal{D}$ is the not the model's training data, but a set of images disjointed from the training data. We sourced images from Fairface [31], opting for faces of children in the age range of 3–9 years, according to their labels. We chose these images for a distribution of natural faces that is different from CelebA's, as it consists of fewer young faces. We experiment with 1000 images as $\mathcal{D}_R$ and 10 images as $\mathcal{D}_F$.

Fig. 6 demonstrates that forgetting is effective, even with different distributions (noticeable difference between Gaussian distributions). Since we use data from a distribution that is different than the training, it is important to consider the similarity between these distributions. In the Supplementary we analyze this effect.

5. Ablation study

We evalute our loss (Eq. (12)) by discarding different parts of it. The base model is fine-tuned to forget 15 images of an identity from CelebA (see Sec. 4.1).

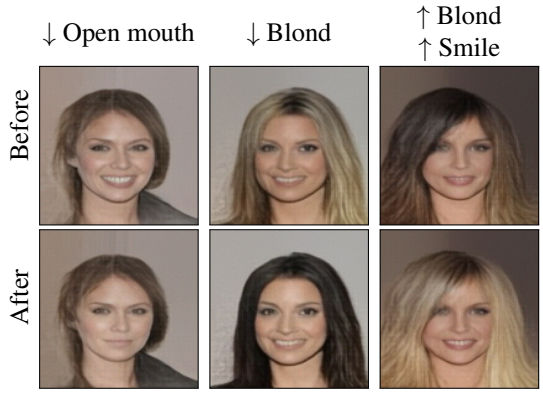

Figure 5. **Change attributes of constant latent vectors.** We sample the same latent vectors and pass them through the base model θ_B (before) and the tamed model θ_T (after) when changing an attribute(s). We see the desired attribute(s) change. Additional examples (including videos) in the Supplementary.

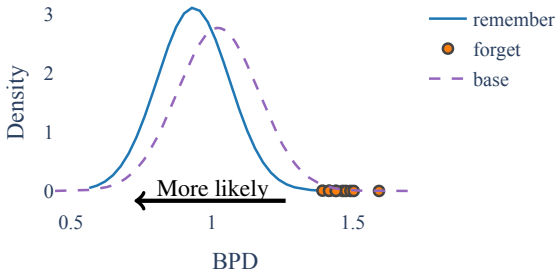

Figure 6. **Forget without training data access.** When the remember images $\mathcal{D}_R$ (solid) differ from the training data of the base model θ_B (dashed), we still forget images w.r.t. the remember images NLL (orange dots). The x-axis is NLL in bits per dimension units (BPD [45]), meaning lower is more likely.

Qualitative comparison

In Fig. 7, we compare the quality of images generated by the different models. We randomly sampled two latent vectors from the prior distribution and passed them through the different models. Without any loss that preserves the knowledge of the original objective (the $\mathcal{L}_R$ loss), the quality of the generated images is significantly worse. Furthermore,

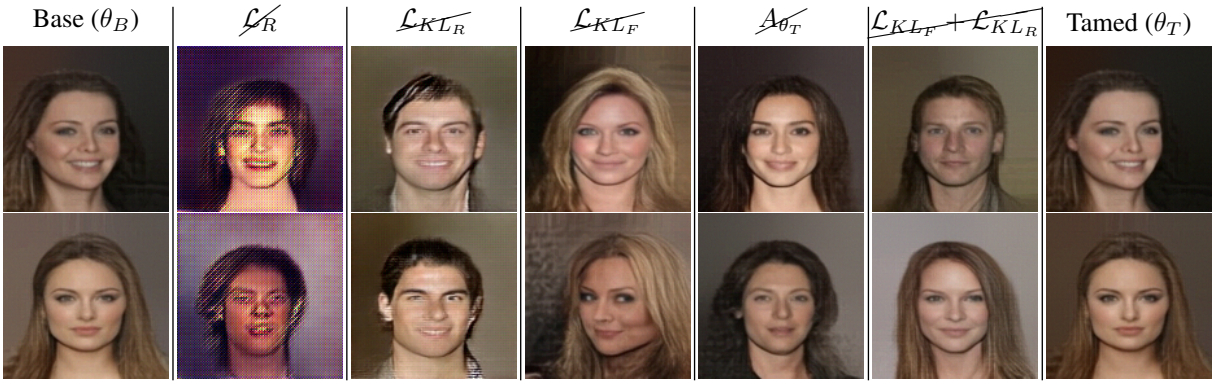

Figure 7. **Qualitative ablation comparison.** For a fixed couple of latent vectors drawn from the tractable prior distribution, the shown images were acquired by passing the latents through different normalizing flows. Each column represents the output of a different model. Each middle column represents a model that was trained while eliminating a different part of the loss, *e.g.* $\not{\mathcal{L}_R}$ uses the entire loss term apart from the remember loss $\mathcal{L}_R$. The different loss components are presented in Eqs. (11) and (12).

we see that the reverse KL divergence loss, $\mathcal{L}_{KL_R}$, is vital to produce images with high quality. While some parts of the loss seem to have less effect on generation quality (columns 4–6 in Fig. 7), only with the full objective do we get a model that preserves the original images. This is evident from the figure, showing that only the rightmost column preserves the images generated using θ_B. This strengthens the assumption that taming preserves the latent space structure, as discussed in Sec. 4.2, and demonstrated in Fig. 5.

Quantitative comparison

Tab. 2 shows a comparison of the different ablated models. A full comparison can be found in the Supplementary. We see that the reverse KL divergence is crucial to ensure a high likelihood of the training data (see BPD), as it is suited for generation tasks. When omitting $\mathcal{L}_R$, the FID [27] score grows (worse), and the forgetting objective (the likelihood quantile column) grows as well. The growth in FID score means we drift from the original data, and the large likelihood score means the model does not forget the data it was supposed to forget. Omitting the $\mathcal{L}_{KL_R}$ term maintains similarity with the original data (low FID), but fails to forget.

Limitations

Our method is demonstrated only on normalizing flows and our precise evaluation is based on the NLL distribution normality assumption. Although we do offer a qualitative measurement using a normality test as mentioned in Sec. 3.3, models that do not align with this assumption will find the threshold we use less powerful and accurate.

6. Ethical considerations

Generative models gained immense popularity in recent years, leading to increased public interest. This raises important issues regarding the generation of hateful, fake, explicit and biased content. Entities that train these models are concerned about the potential risks and opt out of their public release. We propose a direction that can help moderate these malicious applications, and help with their mitigation. Although our method can be used in a positive manner, it can also be used in the opposite direction, *e.g.* increasing a model's bias instead of debiasing it.

Model	BPD mean (↓)	FID(↓)	Forget threshold	Likelihood quantile $\mathrm{q}_\theta(\mathcal{D}_F)$
Base (θ_B)	1.021	140.89	-	-
$\not{\mathcal{L}_R}$	25.640	181.26	✗	0.34
$\not{\mathcal{L}_{KL_R}}$	2.254	**110.34**	✗	0.73
$\not{\mathcal{L}_{KL_F}}$	1.072	121.45	✓	2.45e−5
$\not{A_{\theta_T}}$	1.025	141.99	✓	1.02e−5
$\not{\mathcal{L}_{KL_F} + \mathcal{L}_{KL_R}}$	1.028	143.16	✓	8.53e−6
Tamed (θ_T)	**1.020**	141.63	✓	2.35e−5

Table 2. **Quantitative ablation study.** Removing parts of our loss affects the NLL of $\mathcal{D}_R$ (in BPD [45] units), the generation quality (FID), and the forgetness of $\mathcal{D}_F$, in terms of achieving the threshold (Eq. (10)) and the likelihood quantile $\mathrm{q}_\theta(\mathcal{D}_F)$. For models names notation see Fig. 7.

7. Conclusion

In this work, we proposed an approach towards taming normalizing flow models, controlling their output by increasing or decreasing the probability of generating specific data. We demonstrated different aspects of taming on human faces, showing how to change generation probability both locally and globally. Taming provides an easy modification tool, with minimal collateral damage to the model. Although taming is demonstrated only on normalizing flows, our approach can be extended to other generative models based on exact likelihood estimation.

Acknowledgements This project was supported by ISF grants No. 1574/21 and 1549/19.

References

[1] David Ahmedt-Aristizabal, Mohammad Ali Armin, Simon Denman, Clinton Fookes, and Lars Petersson. Graph-based deep learning for medical diagnosis and analysis: past, present and future. *Sensors*, 21(14):4758, 2021. 2

[2] Shun-ichi Amari. Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196, 1993. 4, 5

[3] Dmitrii Babaev, Maxim Savchenko, Alexander Tuzhilin, and Dmitrii Umerenkov. Et-rnn: Applying deep learning to credit loan applications. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2183–2190, 2019. 2

[4] Sina Baharlouei, Maher Nouiehed, Ahmad Beirami, and Meisam Razaviyayn. R\'enyi fair inference. *arXiv preprint arXiv:1906.12005*, 2019. 2

[5] David Bau, Steven Liu, Tongzhou Wang, Jun-Yan Zhu, and Antonio Torralba. Rewriting a deep generative model. In *European conference on computer vision*, pages 351–369. Springer, 2020. 2

[6] Steffen Bickel, Michael Brückner, and Tobias Scheffer. Discriminative learning under covariate shift. *Journal of Machine Learning Research*, 10(9), 2009. 2

[7] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021. 2

[8] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramèr, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. *arXiv preprint arXiv:2301.13188*, 2023. 1, 2

[9] Nicholas Carlini, Florian Tramèr, Eric Wallace, Matthew Jagielski, Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Úlfar Erlingsson, Alina Oprea, and Colin Raffel. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650. USENIX Association, Aug. 2021. 1, 2

[10] Jianfei Chen, Cheng Lu, Biqi Chenli, Jun Zhu, and Tian Tian. Vflow: More expressive generative flows with variational data augmentation. In *International Conference on Machine Learning*, pages 1660–1669. PMLR, 2020. 3

[11] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When machine unlearning jeopardizes privacy. In *ACM SIGSAC Conference on Computer and Communications Security (CCS)*, 2021. 2

[12] Anton Cherepkov, Andrey Voynov, and Artem Babenko. Navigating the gan parameter space for semantic image editing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3671–3680, 2021. 2

[13] Kristy Choi, Aditya Grover, Trisha Singh, Rui Shu, and Stefano Ermon. Fair generative modeling via weak supervision. In *International Conference on Machine Learning*, pages 1887–1898. PMLR, 2020. 2

[14] The Conversation. https://theconversation.com/the-use-of-deepfakes-can-sow-doubt-creating-confusion-and-distrust-in-viewers-182108. 1

[15] Laurent Dinh, David Krueger, and Yoshua Bengio. NICE: Non-linear Independent Components Estimation, Apr. 2015. Number: arXiv:1410.8516 arXiv:1410.8516 [cs]. 1, 3

[16] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using Real NVP, Feb. 2017. Number: arXiv:1605.08803 arXiv:1605.08803 [cs, stat]. 1, 3

[17] Charles Elkan. The foundations of cost-sensitive learning. In *International joint conference on artificial intelligence*, volume 17, pages 973–978. Lawrence Erlbaum Associates Ltd, 2001. 2

[18] Aditya Golatkar, Alessandro Achille, and Stefano Soatto. Eternal Sunshine of the Spotless Net: Selective Forgetting in Deep Networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9301–9309, Seattle, WA, USA, June 2020. IEEE. 2

[19] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 1, 2

[20] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013. 2

[21] Will Grathwohl, Ricky TQ Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. Ffjord: Free-form continuous dynamics for scalable reversible generative models. *arXiv preprint arXiv:1810.01367*, 2018. 3

[22] Matej Grcić, Ivan Grubišić, and Siniša Šegvić. Densely connected normalizing flows. *Advances in Neural Information Processing Systems*, 34:23968–23982, 2021. 3

[23] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens van der Maaten. Certified Data Removal from Machine Learning Models, Aug. 2020. arXiv:1911.03030 [cs, stat]. 2

[24] Niv Haim, Gal Vardi, Gilad Yehudai, Ohad Shamir, and Michal Irani. Reconstructing training data from trained neural networks. *arXiv preprint arXiv:2206.07758*, 2022. 2

[25] Jun Han and Claudio Moraga. The influence of the sigmoid function parameters on the speed of backpropagation learning. In *International workshop on artificial neural networks*, pages 195–201. Springer, 1995. 4

[26] GM Harshvardhan, Mahendra Kumar Gourisaria, Manjusha Pandey, and Siddharth Swarup Rautaray. A comprehensive survey and analysis of generative models in machine learning. *Computer Science Review*, 38:100285, 2020. 1

[27] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 8

[28] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative

models with variational dequantization and architecture design. In *International Conference on Machine Learning*, pages 2722–2730. PMLR, 2019. 3

[29] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. 1, 2

[30] Cemre Efe Karakas, Alara Dirik, Eylül Yalçınkaya, and Pinar Yanardag. Fairstyle: Debiasing stylegan2 with style channel manipulations. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XIII*, pages 570–586. Springer, 2022. 2

[31] Kimmo Kärkkäinen and Jungseock Joo. Fairface: Face attribute dataset for balanced race, gender, and age. *arXiv preprint arXiv:1908.04913*, 2019. 7

[32] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017. 3

[33] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3, 5

[34] Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative Flow with Invertible 1x1 Convolutions, July 2018. Number: arXiv:1807.03039 arXiv:1807.03039 [cs, stat]. 1, 3, 5

[35] Diederik P Kingma and Max Welling. Stochastic gradient vb and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*, volume 19, page 121, 2014. 1, 2

[36] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proc. Natl. Acad. Sci. U.S.A.*, 114(13):3521–3526, Mar. 2017. arXiv:1612.00796 [cs, stat]. 2

[37] Ivan Kobyzev, Simon JD Prince, and Marcus A Brubaker. Normalizing flows: An introduction and review of current methods. *IEEE transactions on pattern analysis and machine intelligence*, 43(11):3964–3979, 2020. 1, 3

[38] Zhifeng Kong and Kamalika Chaudhuri. Forgetting data from pre-trained gans. *arXiv preprint arXiv:2206.14389*, 2022. 2

[39] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951. 5

[40] Lucien Le Cam. Maximum likelihood: an introduction. *International Statistical Review/Revue Internationale de Statistique*, pages 153–171, 1990. 2

[41] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015. 1, 5

[42] F. J. Massey. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American Statistical Association*, 46(253):68–78, 1951. 4

[43] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Icml*, 2010. 1

[44] OUTRIDER. https://outrider.org/nuclear-weapons/articles/could-chatbot-teach-you-how-build-dirty-bomb. 2

[45] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked autoregressive flow for density estimation. *Advances in neural information processing systems*, 30, 2017. 7, 8

[46] The Washington Post. https://www.washingtonpost.com/nation/2021/03/13/cheer-mom-deepfake-teammates/. 1

[47] Vikram V Ramaswamy, Sunnie SY Kim, and Olga Russakovsky. Fair attribute classification through latent space de-biasing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9301–9310, 2021. 2

[48] General Data Protection Regulation. https://gdpr-info.eu/. 1, 2

[49] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. 1, 2

[50] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015. 3

[51] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 7

[52] Alon Shoshan, Nadav Bhonker, Igor Kviatkovsky, and Gerard Medioni. Gan-control: Explicitly controllable gans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14083–14093, 2021. 7

[53] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 2

[54] Shuhan Tan, Yujun Shen, and Bolei Zhou. Improving the fairness of deep generative models without retraining. *arXiv preprint arXiv:2012.04842*, 2020. 2

[55] Ryutaro Tanno, Melanie F Pradier, Aditya Nori, and Yingzhen Li. Repairing neural networks by leaving the right past behind. *arXiv preprint arXiv:2207.04806*, 2022. 2

[56] Harri Valpola, Xavier Giannakopoulos, Antti Honkela, and Juha Karhunen. Nonlinear independent component analysis using ensemble learning: Experiments and discussion. In *Proc. Int. Workshop on Independent Component Analysis and Blind Signal Separation (ICA2000)*, pages 351–356, 2000. 2

[57] Aäron van den Oord and Nal Kalchbrenner. Pixel rnn. In *ICML*, 2016. 2

[58] Sheng-Yu Wang, David Bau, and Jun-Yan Zhu. Rewriting geometric rules of a gan. *ACM Transactions on Graphics (TOG)*, 41(4):1–16, 2022. 2

[59] Tianlu Wang, Jieyu Zhao, Mark Yatskar, Kai-Wei Chang, and Vicente Ordonez. Balanced datasets are not enough: Estimating and mitigating gender bias in deep image representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5310–5319, 2019. 2

[60] WIRED. https://www.wired.com/story/zelensky-deepfake-facebook-twitter-playbook/. 1

[61] Chen Henry Wu, Saman Motamed, Shaunak Srivastava, and Fernando De la Torre. Generative visual prompt: Unifying distributional control of pre-trained generative models. *arXiv preprint arXiv:2209.06970*, 2022. 2

[62] Ga Wu, Masoud Hashemi, and Christopher Srinivasa. Puma: Performance unchanged model augmentation for training data removal. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8675–8682, 2022. 2

[63] Brian Hu Zhang, Blake Lemoine, and Margaret Mitchell. Mitigating unwanted biases with adversarial learning. In *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pages 335–340, 2018. 2