Self-Cleaning Diffusion Models

Adrian Rodriguez-Munoz[1]* **Adam Klivans**[2] **Antonio Torralba**[1]
Constantinos Daskalakis[1] **Giannis Daras**[1]

[1]MIT, Department of Electrical Engineering and Computer Science
[2]UT Austin, Department of Computer Science

Abstract

We present Self-Cleaning Diffusion, a principled and effective framework for training diffusion models from heterogeneous data. In many domains, high-quality data is scarce or expensive to collect, while low-quality and out-of-distribution (OOD) data is abundant. Leveraging these ubiquitous samples is critical for scaling generative models. Still, principled techniques remain elusive, leaving practitioners to rely on simple heuristics such as high-quality finetuning or explicit quality conditioning. Self-Cleaning Diffusion bridges this gap by decoupling data correction from prior learning. We first train a transport map between the abundant OOD and the limited in-distribution samples. We then train an unconditional generative prior on a mixture of the few in-distribution samples and *noisy* versions of the transported points. Crucially, this noise injection prevents learning errors in the transport map from propagating to the generative prior. Theoretically, we demonstrate that transporting OOD samples towards the empirical proxy reduces the distance to the underlying distribution under mild assumptions. Empirically, we achieve state-of-the-art results across a diverse range of settings, from mitigating synthetic corruptions to increasing diversity and transcending dataset quality.

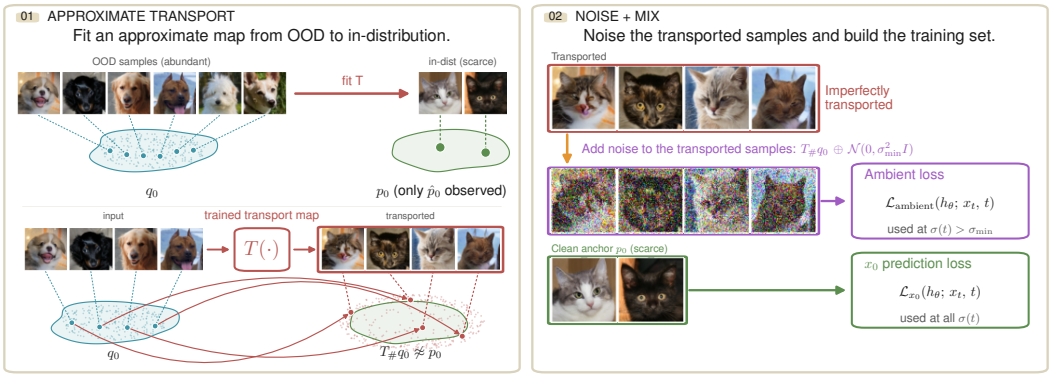

Figure 1: Visual illustration of our algorithm. In stage 1, we fit a transport map T between a large set of OOD samples (blue points) and a small set of in-distribution samples (green points). The transported points often land outside of the target manifold since the map is only trained with limited clean targets. In Stage 2, we add noise to the transported samples to "erase" mistakes made by the transport map. The combined set of the limited clean samples and the *noisy* transformed samples becomes the training dataset for our diffusion model. The transported datapoints only contribute in the high-noise regime of the diffusion process.

*Correspondence to: adrianrm@mit.edu.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

1 Introduction

The impressive success of the Generative AI field over the last couple of years is attributed vastly to scale; of compute, parameters and data [30, 34, 39]. From these three axes of scaling, data is becoming perhaps the most critical resource [76]. Indeed, in many applications, especially in science, collecting large-scale high-quality datasets is challenging, posing significant roadblocks to the wider adoption of Generative AI in such fields [17, 77]. While collecting high-quality data is expensive (or sometimes impossible), low-quality or out-of-distribution data is cheaper and widely available.

Recently, there has been an explosive growth of interest in developing principled methods to leverage imperfect data during training effectively [1, 4, 6, 17, 18, 19, 35, 42, 48, 49, 51, 53, 57, 61, 63, 69]. Despite the increased interest in this topic, the proposed methods are not widely adopted in training pipelines for generative modeling. Instead, commercial models often rely on simple heuristics to deal with the heterogeneity of the training samples, including aggressive filtering methods [27, 44, 67], high-quality finetuning [12, 24, 58], explicitly conditioning the model on sample quality [9, 58, 81], loss or gradient reweighting [56], and data-mixing [24, 27]. A potential reason for this lack of broader adoption is that often the developed algorithms suffer from (at least) one of the following issues: (a) they assume knowledge of the degradation process [4, 19, 35], (b) they are too wasteful [17, 18] (essentially reverting to filtering of the out-of-distribution data), (c) they add significant computational overhead to the training process [4, 13, 16, 35, 61].

To bridge the gap between principled theory and scalable practice, we propose Self-Cleaning Diffusion: a computationally efficient framework for training diffusion models using a small anchor of in-distribution data and a massive corpus of imperfect, synthetic, or domain-shifted data. Self-Cleaning Diffusion operates on a simple but powerful premise: we decouple data-cleaning from the training of the generative prior. Our framework consists of two straightforward stages. First, we train a transport map to project the abundant OOD data into the in-distribution space. Second, we train an unconditional diffusion model on a mixture of the initial in-distribution samples and *noisy* versions of the transported samples. Crucially, injecting noise accounts for learning errors in the transport map, preventing error propagation across stages [2, 3]. This decoupled approach yields distinct theoretical advantages. Learning solely from a limited in-distribution dataset incurs high statistical error, while directly mixing in OOD samples induces severe sampling bias. Our methodology improves upon this trade-off: it reduces statistical error by effectively expanding the sample size, while actively shrinking bias by transporting the OOD samples prior to generative modeling.

To demonstrate the efficacy of Self-Cleaning Diffusion, we comprehensively validate our framework across a diverse spectrum of experimental scenarios. Our evaluations span from mitigating synthetic data corruptions, such as fog and blur applied to ImageNet, to utilizing data with severe out-of-distribution shifts. Furthermore, we show that our framework can transcend the image quality of datasets without relying on any additional pristine data. Across all these settings, our approach consistently outperforms prior state-of-the-art techniques for learning from heterogeneous data.

2 Background and Related Work

2.1 Diffusion Modeling

Diffusion is a prevailing framework for learning to sample from continuous, high-dimensional distributions [32, 72]. The core idea is to reduce the problem of sampling from a distribution to an estimation problem that can be solved with supervised learning. In particular, in diffusion modeling, we are given access to samples $X_0 \sim p_0$ from the target distribution p_0 and we form noisy iterates $X_t = X_0 + \sigma(t)Z, \quad Z \sim \mathcal{N}(0, I_d)$, for different times t sampled uniformly in $[0, T]$, where T is a large constant and $\sigma(t)$ is an increasing function that determines the standard deviation of the noise at each time t. The goal then is to learn a function h_θ, parametrized by $\theta \in \Theta$, that optimally denoises the noisy samples, using the following objective[2]:

$$J_{x_0}(\theta) = \mathbb{E}_{(X_0,t,X_t)}[||h_\theta(X_t, t) - X_0||^2]. \tag{1}$$

[2]There are several parametrizations of this objective, that often lead to superior results and faster sampling [40, 47, 65].

For any noise level t, the learned function $h_\theta(\cdot, t)$ is trained to approximate the conditional expectation $\mathbb{E}[X_0|X_t = \cdot, t]$, which is the optimal denoiser in the l_2-sense. A beautiful result from stochastic calculus guarantees that having access to the family of conditional expectations $\{\mathbb{E}[X_0|X_t = \cdot, t]\}_{t=0}^{\infty}$ suffices in order to sample from p_0 through a stochastic differential equation [10, 54, 73]. A deterministic sampling process with similar sampling guarantees was established in [73].

2.2 Learning Diffusion Models from Corrupted Data

In the discussion so far, we have assumed access to infinite number of samples from the target density p_0. However, in practice, we are typically dealing with a finite dataset of samples, $\mathcal{D}_{p_0} = \{X_0^{(i)}\}_{i=1}^n$. If the number of samples from the target distribution p_0 is small, then optimizing over this finite dataset leads to a poor estimation of the underlying continuous density [37, 69]. Hence, a plethora of recent works [1, 4, 17, 18, 19, 35, 42, 48, 49, 51, 53, 57, 61, 63, 69] attempt to leverage additional samples $\mathcal{D}_{q_0} = \{Y_0^{(i)}\}_{i=1}^m$, that are distributed according to some other distribution q_0[3].

The majority of the proposed methods assume that the relationship between the two distributions, p_0 and q_0, is known and simple. For example, in Ambient Diffusion [19], the authors show how to train diffusion models from linearly corrupted data, in Consistent Diffusion Meets Tweedie [16] from samples with additive Gaussian noise, and there are Expectation-Maximization based approaches that can deal with any known and differentiable degradation channel [4, 35, 63].

However, in most practical settings, the corruption process is not given, and q_0 can just represent some unknown distribution shift that appears naturally. To make this argument concrete, we can think of p_0 as the distribution of natural images captured by professional photographers and q_0 as the distribution of natural images captured by ordinary people. While there is definitely semantic overlap and an opportunity for learning from q_0 samples, a mathematical relationship between these two distributions is not known.

2.2.1 Ambient Diffusion Omni

A general approach to tackle this problem was proposed in Ambient Diffusion Omni [18]. The key idea is that the addition of noise contracts distributional shifts, so for high-enough noise samples from $q_t = q_0 \circledast \mathcal{N}(0, \sigma^2(t) I_d)$ approximate well samples from $p_t = p_0 \circledast \mathcal{N}(0, \sigma^2(t) I_d)$. We denote with $\sigma(t_{\min})$ the smallest noise level for which $q_{t_{\min}} \approx p_{t_{\min}}$. For all $t : \sigma(t) > \sigma(t_{\min})$, the authors of [18] can use corrupted data $Y_0 \sim q_0$ to approximate $\mathbb{E}[X_0|X_t = \cdot, t]$ using the following objective:

$$J_{\text{ambient}}(\theta, Y_{t_{\min}}, t_{\min}, t) = \mathbb{E}_{Y_t|Y_{t_{\min}}}$$
$$\left[\left|\left|\alpha(t, t_{\min}) h_\theta(Y_t, t) + (1 - \alpha(t, t_{\min}))Y_t - Y_{t_{\min}}\right|\right|^2\right], \tag{2}$$

where $\alpha(t, t_{\min}) = \frac{\sigma^2(t) - \sigma^2(t_{\min})}{\sigma^2(t)}$. This objective can only be used safely in the high-noise regime of the diffusion, i.e. for $t : \sigma(t) > \sigma(t_{\min})$, and hence, the utility of this approach depends on how large $\sigma(t_{\min})$ is. Our core insight in this work is that this approach can be very wasteful, especially for low-frequency corruptions. Take a mean-shift, for example. Even though the corruption process is very simple, the amount of (zero-mean) noise that needs to be added to erase it is infinity, rendering samples from q_0 useless for learning. The authors of [18] primarily test their approach against high-frequency corruptions. Due to the spectral power-law of natural images, addition of noise effectively masks the high frequencies of a signal [14, 23, 25, 75]. As a result, signals that are corrupted only in the high-frequency part of the spectrum can be safely used in large parts of the diffusion process without introducing biases. As we show in this work, the method of [18] fails dramatically in low-frequency corruptions and other type of shifts that cannot be easily erased by additive Gaussian noise.

2.2.2 Natural baselines

Despite the lack of theoretical guarantees, practitioners rely on simple heuristics to leverage imperfect samples: high-quality fine-tuning [12, 24, 58], data-mixing [24, 27, 44], and conditioning the model

[3] Samples from q_0 can correspond to low-quality, synthetic or just simply out-of-distribution datapoints. For the rest of the paper, we refer to these samples as *imperfect* or OOD datapoints, and we will be using these notations interchangeably.

on sample quality [9, 58, 81]. We describe them in Appendix B.4 and compare against them in Section 4. Inference-time and RL-based methods for steering the distribution [3, 5, 22, 33, 80] are complementary and can be combined with our findings.

3 Method and Theoretical Results

In this section, we explain the Self-Cleaning Diffusion framework in detail. A high-level introduction of the algorithm is given in Section 3.1. We provide a theoretical analysis in Section 3.2 and we concretize implementation details in Section 3.3.

To introduce some notation, we will be denoting by $\hat{p}_0$ the empirical distribution over in-distribution samples from p_0. Namely, $\hat{p}_0 := \frac{1}{n}\sum_{i=1}^{n}\delta_{X_i}$ where $X_1, \ldots, X_n \overset{\text{i.i.d.}}{\sim} p_0$. For ease of presentation, we assume that we have sample access to q_0, i.e. as many OOD samples as we want. We further assume that p_0 and q_0 are measures in $\mathbb{R}^d$ that have second moments, i.e. belong to $\mathcal{P}_2(\mathbb{R}^d)$. We will also assume that q_0 is absolutely continuous, and that $W_2(q_0, p_0) > 0$, where W_2 denotes the Wasserstein-2 distance between measures.

3.1 Self-Cleaning Diffusion

A visual illustration of our approach is given in Figure 1. It consists of the following conceptual steps:

- **Stage 1: Approximate Transport**. In this step, we fit a transport map $T(\cdot)$ between q_0 and $\hat{p}_0$. We think of this transport map as a conditional generative model that takes as input a sample from q_0 and tries to transform it into a sample of p_0. Unfortunately, we don't have access to p_0 directly, so we fit this transport map to its proxy, $\hat{p}_0$. To avoid the pathological trivial case of fully mapping to $\hat{p}_0$, which we already have, we penalize huge movements from q_0 during the fitting of the transport map. The hope is that even though we are not transporting to the actual distribution p_0, we are making some progress towards p_0 as we move towards its empirical proxy, $\hat{p}_0$. We formalize this intuition in the Theory Section and empirical ways of performing this mapping via Drifting Models [21] in Section 3.3.
- **Stage 2: Accounting for errors in the transport map**. After the transport map is fit, we can get samples from the transported distribution $T\#q_0$. It is reasonable to expect that even though $T\#q_0$ is probably closer to p_0 than q_0 is, there is still a bias due to the limited size of samples we used to train the transport map. To reduce the bias, we use the Ambient Diffusion Omni [18] trick: we noise the samples from $T\#q_0$ with the minimum noise level $t_{\min}$ needed such that: $T\#q_0 \circledast \mathcal{N}(0, \sigma^2(t_{\min})I) \approx p_0 \circledast \mathcal{N}(0, \sigma^2(t_{\min})I)$. Simply put, our approach noises the transported samples instead of the original OOD samples, with the hope that the noise level required to make them indistinguishable from real noisy samples is lower. Once again, we formalize this intuition in Section 3.2.
- **Stage 3: Training with noisy data**. Once the previous stage is complete, we have a dataset that is a mixture of a few clean datapoints ($\hat{p}_0$ distribution) and a large set of samples (transported points under noise) that approximate well $p_{t_{\min}}$. We use the clean samples to train the model for all diffusion levels using the objective of Eq. 1 and the noisy samples for all noise levels $t : \sigma(t) \geq \sigma(t_{\min})$ using the Ambient objective (Eq. 2). This objective can also be applied in a short fine-tuning stage of a model pretrained on all the data (Section 3.3).

3.2 Theoretical Analysis

We start our theoretical analysis by defining a regularized optimal transport problem from distribution q_0 to distribution $\hat{p}_0$, for any $s \in [0, 1]$, as follows:

$$\rho_s^\star \;=\; \arg\min_{\rho} \; \left\{ s W_2^2\big(\rho,\, \hat{p}_0\big) \;+\; (1-s)\, W_2^2\big(\rho,\, q_0\big) \right\}. \tag{3}$$

The first term pulls ρ towards the empirical distribution, while taking $s < 1$ prevents the optimizer to be trivial, i.e. $\rho = \hat{p}_0$. The minimizer $\rho_s^\star$ exists for all $s \in [0, 1]$, because q_0 and $\hat{p}_0$ have finite second moments [8]. It is also unique because q_0 is absolutely continuous. For the same reason, there exists

a (q_0-almost everywhere) unique Brenier map $T : \mathbb{R}^d \to \mathbb{R}^d$ such that for all $s \in [0, 1]$:

$$\rho_s^\star = ((1 - s)\ \mathrm{Id} + s\ T)_{\#} q_0.$$

In what follows, we treat s as a parameter that we control. It is straightforward to see that if we allow ourselves to control s, we can always achieve at least the same distance to p_0 as the best of the endpoints, q_0 or $\hat{p}_0$. Indeed, for $s = 1$ we recover $\hat{p}_0$, while for $s = 0$ we recover q_0. The question is under which conditions can we strictly improve over both endpoints.

We are now ready to present a tool that will help us identify conditions under which moving from q_0 part-way towards the empirical $\hat{p}_0$ produces a distribution closer to p_0.

Theorem 3.1 (Transport towards the empirical reduces distribution distance)**.** *Let T be a Brenier map from q_0 to $\hat{p}_0$ and let Φ be a Brenier map from q_0 to p_0. Define*

$$\beta \ := \ \mathbb{E}_{x \sim q_0} \big\langle T(x) - x,\ x - \Phi(x) \big\rangle.$$

Then for every $s \in [0, 1]$,

$$W_2^2\big(\rho_s^\star,\, p_0\big) - W_2^2(q_0,\, p_0) \ \leq \ s^2 W_2^2\big(q_0,\, \hat{p}_0\big) + 2s\beta. \qquad (4)$$

The proof of this result is given in Appendix A.

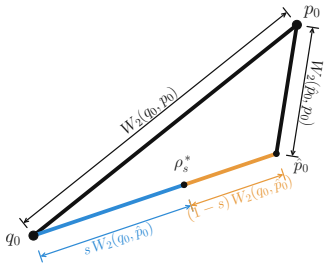

Figure 2: Geometric picture of geodesic paths between the empirical $\hat{p}_0$, the OOD q_0 and the target density p_0.

For $\rho_s^\star$ to have made progress towards the target distribution p_0 compared to q_0, we would like the term on the right-hand side of (4) to be negative. By Cauchy–Schwarz, $|\beta| \leq W_2(q_0, p_0) \cdot W_2(q_0, \hat{p}_0)$, so we may define $\theta \in [0, \pi]$ via the following $\beta = -\cos\theta \cdot W_2(q_0, p_0) \cdot W_2(q_0, \hat{p}_0)$. Speaking imprecisely, θ captures the average angle between $\Phi(x) - x$ and $T(x) - x$, i.e. between how the average $x \sim q_0$ is transported under the optimal transport to p_0 and to $\hat{p}_0$. When these have on average an acute angle, β is negative, and for a good choice of s the right-hand side of (4) is also negative. We gather this into the following corollary, providing simple conditions under which transporting from q_0 towards the proxy $\hat{p}_0$ brings us closer to p_0 compared to q_0.

Corollary 3.2 (Condition for progress over q_0)**.** *If $\cos\theta > 0$, there exists some $s \in (0, 1)$ such that $W_2(q_0, p_0) > W_2(\rho_s^*, p_0)$. Under the stronger condition*

$$\cos(\theta) > \frac{1}{2} \frac{W_2(q_0, \hat{p}_0)}{W_2(q_0, p_0)}, \qquad (5)$$

we have that $W_2(q_0, p_0) > W_2(\rho_s^*, p_0)$ for all $s \in (0, 1)$, and for the condition to not be vacuous we need that $W_2(q_0, p_0) > \frac{1}{2} W_2(q_0, \hat{p}_0)$.

Remark 3.3. Bound (5) implies an upper-bound on the angle θ, i.e. $\theta < \cos^{-1}\left(\frac{1}{2} \frac{W_2(q_0, \hat{p}_0)}{W_2(q_0, p_0)}\right)$, so that all $\rho_s^\star$ on the geodesic from q_0 to $\hat{p}_0$ are closer to p_0 than q_0 is. The need for an upper bound on the angle is intuitive; we need that, on average, the direction we take to map a q_0 sample to a p_0 sample under the optimal coupling of these distributions is not too misaligned with the direction we take to map that same sample from q_0 to a sample of $\hat{p}_0$ under the optimal coupling of those distributions. Hence, transporting towards $\hat{p}_0$ takes us closer to p_0 compared to how far q_0 is from p_0. If we only want to beat q_0 for some s, then all we need is that this angle is at most $\pi/2$.

Under the conditions of Corollary 3.2, we understand when transporting the samples of q_0 towards the proxy of $\hat{p}_0$ is beneficial. But a question that remains is under what conditions the transported distribution $\rho_s^\star$ may achieves a lower distance than the empirical proxy itself. The following corollary identifies sufficient conditions for that to happen.

Corollary 3.4 (Condition for progress over $\hat{p}_0$)**.** *Assume WLOG $W_2(\hat{p}_0, p_0) \leq W_2(q_0, p_0)$. If the following bound holds:*

$$\frac{\sqrt{W_2^2(q_0, p_0) - W_2^2(\hat{p}_0, p_0)}}{W_2(q_0, p_0)} < \cos(\theta) < \frac{W_2(q_0, \hat{p}_0)}{W_2(q_0, p_0)}, \qquad (6)$$

then there is an $s^* \in (0, 1)$ such that: $W_2(\rho_{s^*}^*, p_0) < W_2(\hat{p}_0, p_0) = \min\{W_2(\hat{p}_0, p_0), W_2(q_0, p_0)\}$. For (6) to not be vacuous we need that $W_2^2(q_0, p_0) \leq W_2^2(\hat{p}_0, p_0) + W_2^2(q_0, \hat{p}_0)$.

Remark 3.5 (Geometric interpretation of strict improvement). The result above has the following interpretation:

- **Lower bound (Alignment):** The transport map to $\hat{p}_0$ must point sufficiently close to the transport map to p_0. If the angle θ is too wide (i.e., $\cos\theta$ is too small), the interpolation path diverges and the minimum distance fails to drop below the cost of the empirical distribution $W_2(\hat{p}_0, p_0)$.
- **Upper bound (No overshooting):** The transport map to $\hat{p}_0$ cannot be too perfectly aligned relative to the distances involved. If $\cos\theta$ is too large, the optimal unconstrained step size s_{free} exceeds 1. Because the interpolant is constrained to $s \in [0, 1]$, the minimizer gets clamped at $s^\star = 1$, resulting in the exact reconstruction of $\hat{p}_0$ without strict improvement.

Remark 3.6 (From W_2 to FID and DINOv2 FD). FID and DINOv2 FD are the squared W_2 distance between Gaussians fitted to the Inception-v3 or DINOv2 features $\phi(x)$ of two distributions. Fitting Gaussians cannot increase W_2 [29], so if ϕ is L-Lipschitz, $\text{FD}_\phi(p, q) \leq W_2^2(\phi_\# p, \phi_\# q) \leq L^2\, W_2^2(p, q)$ (Appendix A.2). The bounds of Theorem 3.1 and Corollaries 3.2 and 3.4 thus carry over to the FID and DINOv2 FD of $\rho_s^\star$ up to the factor L^2.

3.3 Practical considerations

Implementations of the transport map. The algorithm in Section 3.1 requires a way to map between the OOD samples and the in-distribution samples from p_0, with the asymmetric aspect being that we have access to a lot more OOD samples than in-distribution samples. Our theoretical formulation further commits to a particular objective to be minimized when training this transport map, namely the objective of Eq. (3). That said, even simple, imperfect ways of bringing the samples closer to p_0 should help, as we can always protect from mistakes of the transport map by adding more noise in Stage 2 of our algorithm (at the expense of being more wasteful with how we use the OOD samples).

Over the past few years, multiple methods have been proposed to map between two distributions without pairs, with notable examples being CycleGAN [82], CycleDiffusion [78], Schrodinger Bridges [7, 43, 46], flow-matching [45, 47, 52], and, more recently, drifting models [21]. In this paper, we use conditional drifting models to perform this approximate mapping, and we compare them with rectified flows and CycleGAN in Section 4.4. We provide a detailed overview of Drifting models and how we use them to fit the transport map in Appendix Section B.5. Our analysis concerns the regularized optimal transport map of Eq. (3), whose properties we can characterize, unlike those of a map learned by gradient descent on the drifting objective; we treat the latter as a practical, and possibly crude, approximation of the former.

Controlling (over-) fitting to $\hat{p}_0$. With limited samples from p_0, the transport map can overfit to $\hat{p}_0$ instead of making progress towards p_0, as generative models memorize small training sets [37, 38, 50, 60, 69, 70, 71]. Our analysis controls this through the parameter s of Eq. (3). In practice, conditional models like ours collapse much less to the target than unconditional ones, we only observed overfitting to $\hat{p}_0$ when clean data was extremely scarce, and it can be mitigated by the model size, early stopping or data augmentation (Appendix B.3).

When does transport help? The benefit of our method depends on (a) the number of clean samples, (b) the distance between q_0 and p_0, (c) the transport family, and (d) $\sigma(t_{\min})$. More clean samples and smaller shifts give a more accurate map (Section 4.4 ablates (a)–(c)), and noise masks the remaining errors, so a worse map needs a larger $\sigma(t_{\min})$, up to discarding the OOD samples. Factor (b) is the hardest to quantify in advance: it is unclear, *e.g.*, whether dogs or foxes are closer to cats.

Selecting $\sigma(t_{\min})$. Perhaps the most important hyperparameter in our method is the amount of noise $\sigma_{\min} := \sigma(t_{\min})$ added to the transported samples: setting it too high wastes the OOD samples, and setting it too low propagates errors of the transport map to the diffusion prior. In our experiments, we sweep it by training a model for each candidate value (Figure 4), which is expensive. A cheap alternative selects it at inference time with two networks that standard pipelines already train (Algorithm 1): a model pretrained on all the data and its fine-tune on the clean samples (HQ fine-tuning). Sampling with the former above a threshold σ_{th} and with the latter below it stands in for a model trained with $\sigma_{\min} = \sigma_{\text{th}}$. One then picks the best threshold on a grid and fine-tunes the pretrained model at this $\sigma_{\min}$, which adds only the transport, the sweep and one fine-tuning run to a standard pipeline of pretraining and HQ fine-tuning. On blur, this sweep selects the same $\sigma_{\min}$

Algorithm 1 Self-Cleaning Diffusion training, including the selection of $\sigma_{\min}$.

Input: clean samples $\mathcal{D}_{p_0}$, OOD samples $\mathcal{D}_{q_0}$, grid $\mathcal{S}$ of candidate thresholds, including 0 and ∞.

1. **Transport:** train a conditional drifting model T from $\mathcal{D}_{q_0}$ to $\mathcal{D}_{p_0}$; $\mathcal{D}_T \leftarrow \{T(y) : y \in \mathcal{D}_{q_0}\}$.
2. **Pretrain:** train θ_{pre} on $\mathcal{D}_{p_0} \cup \mathcal{D}_T$ (Eq. 1).
3. **HQ fine-tune:** $\theta_{\mathrm{hq}} \leftarrow$ fine-tune θ_{pre} on $\mathcal{D}_{p_0}$.
4. **Sweep:** for each $\sigma_{\mathrm{th}} \in \mathcal{S}$, generate samples with θ_{pre} at noise levels $\sigma > \sigma_{\mathrm{th}}$ and θ_{hq} at $\sigma \leq \sigma_{\mathrm{th}}$; set $\sigma_{\min}$ to the σ_{th} with the best metric (*e.g.*, the lowest FID).
5. **Ambient fine-tune:** $\theta \leftarrow$ fine-tune θ_{pre} as in step 3, on $\mathcal{D}_{p_0}$ at all noise levels (Eq. 1) and on $\mathcal{D}_T$ only at $\sigma > \sigma_{\min}$ (Eq. 2).

Output: θ.

as training for both FID and DINOv2 FD, with Spearman rank correlations of 0.7 and 1.0 to the trained models (Appendix E.1). The classifier-based estimate of [18], the lowest noise level at which a noise-conditional classifier confuses clean and OOD samples, was too high in our experiments.

4 Experiments

We now proceed to experimentally validate our method. We try three different sets of experiments: removing synthetic corruptions from ImageNet (Section 4.1), leveraging data from a different class (Section 4.2), and increasing the quality of a generative model without additional data (Section 4.3). We also ablate the transport stage (Section 4.4). In the ImageNet experiments, we select $\sigma_{\min}$ by training a model for each candidate value; Appendix E.1 shows that the cheaper two-network inference-based sweep of Algorithm 1 selects the same values on blur.

4.1 Learning generative models with synthetic corruptions

We begin with synthetic corruptions to test our approach and gain some intuition. Specifically, we take the ImageNet training set and artificially corrupt a very large part of it (90%) with some degradation function. The goal is to use the mixture of 90% degraded ImageNet and 10% clean ImageNet and get a strong generative model that approximates the performance of training on 100% clean ImageNet, without assuming any knowledge of the degradation function.

Baselines. We compare our method with four baselines: (1) the filtering approach, which only trains on the 10% clean data, (2) the class-conditional approach, which trains on all data conditioning on the corruption label and uses only the clean label during inference, (3) the fine-tuning approach (HQ fine-tuning), which pre-trains on all data then fine-tunes on the filtered subset, (4) and finally Ambient Omni [18]. We also include the skyline for training on 100% clean data for reference.

Low frequency fog corruption. The first corruption we try is fog corruption. Specifically, the degradation operator blends a black-and-white fractal pattern with the image affecting its mean pixel statistics. Examples of this corruption are given in Figure 3 (Row 1, left). Results for all baselines and our approach are given in Table 1 and visual generations for the same seeds in Figure 3.

Starting with the filtering baseline, we observe that it achieves a high FID score (large distance to the target distribution), the highest memorization (23% of the images are memorized), and the lowest diversity (Vendi). Figure 3 further shows that the filtering baseline produces low-quality images. Ambient Diffusion Omni [18] achieves the exact same score as the filtering approach, confirming our intuition that this method is not suitable for low-frequency corruptions: noise masks the fog only as it approaches infinity, where Omni discards the foggy images and reduces to filtering; at lower noise, it generates fog (Figure 3, row 3, right). Indeed, after sweeping the parameter $\sigma_{\min}$ for Omni, we found that for the optimal noise level, the fog-images are used by less than 0.001% of the diffusion updates. The two heuristic baselines, high-quality finetuning and conditioning on the corruption label, achieve remarkably stronger results.

Yet, our approach, Self-Cleaning Diffusion, achieves even better results, outperforming the baselines by a large margin. We further note that the optimal $\sigma_{\min}$ for this corruption level is 0.397, confirming that both the transport map (Stage 1) and the noising (Stage 2) are essential for our algorithm. Figure 3 (left) shows that the transport map gets rid of the fog corruption but introduces mild image artifacts. The noising process erases those artifacts, allowing the model to learn better. The transported data have a much better FID than the foggy data because fog destroys little information about the clean

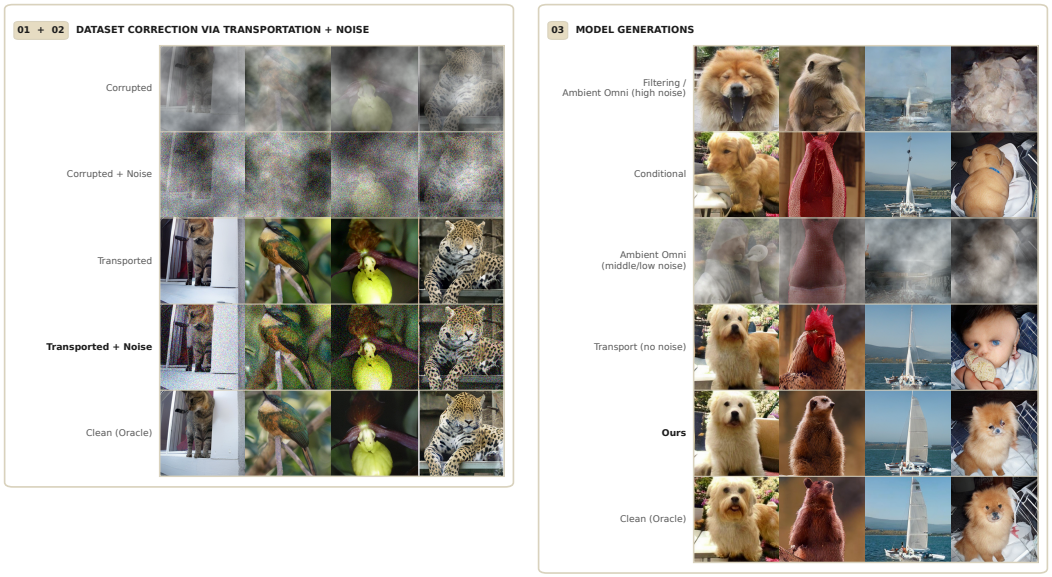

Figure 3: Visualization of dataset correction (left) and generations by different approaches for the same seeds (right) in the 90% fog setting. Transported images (left, row 3) exhibit artifacts, which we "erase" using noise (left, row 4); the same noise does not mask the fog (left, row 2). The FID of the corrupted data is 64.35, that of the transported data 3.86. Ambient Omni reduces to filtering at high noise (right, row 1) and generates fog at middle or low noise (right, row 3). Self-Cleaning Diffusion achieves better generations than the other methods shown.

image but perturbs its features strongly (Appendix E.2). Remarkably, Self-Cleaning Diffusion is only within 0.06 FID points from the clean training oracle. Visual generations in Figure 3 (right) further show that for the same seed, our method and the clean oracle produce very close generations, which implies that the true underlying score function is well approximated.

Table 1: Performance comparison for the fog corruption (90%) setting. FID and DINOv2 FD: mean $\pm$ standard deviation over sets of generated samples with different seeds (Appendix D.6). Self-Cleaning Diffusion + fine-tuning: the $\sigma_{\min} = 0.186$ model fine-tuned with $\sigma_{\min} = 0.397$.

Method	$\sigma_{\min}$	FID ↓	Val FID ↓	DINOv2 FD ↓	DINOv2 FD Val ↓	Memorization (sim > 0.8) ↓	Vendi Score ↑
Filtering	∞	72.48 ± 0.25	70.08 ± 0.27	742.74 ± 1.91	748.84 ± 2.02	0.2362	212.99
HQ Finetuning	∞	18.73 ± 0.05	17.37 ± 0.04	330.70 ± 1.02	341.52 ± 1.02	0.1286	544.90
Conditional	0.000	15.38 ± 0.05	13.37 ± 0.04	353.21 ± 1.08	361.50 ± 1.17	**0.1020**	512.14
Ambient Omni	$\to \infty$	72.48 ± 0.25	70.08 ± 0.27	742.74 ± 1.91	748.84 ± 2.02	0.2362	212.99
Self-Cleaning Diffusion (Ours)	0.186	11.25 ± 0.03	10.01 ± 0.01	353.46 ± 1.15	363.07 ± 1.05	0.1240	498.92
Self-Cleaning Diffusion + fine-tuning (Ours)	0.397	$\mathbf{10.19 \pm 0.07}$	$\mathbf{9.39 \pm 0.07}$	$\mathbf{288.15 \pm 0.83}$	$\mathbf{299.12 \pm 0.88}$	0.1684	**546.80**
All clean	0.000	10.13 ± 0.06	9.73 ± 0.06	219.43 ± 0.58	231.49 ± 0.57	0.2196	608.39

High frequency blur corruption. Having established the superiority of our approach over Ambient Diffusion Omni in low-frequency corruptions, we now target the Gaussian blur corruption, which eliminates high-frequency details but does not affect structural properties of the image. To our surprise, our method outperforms [18] in this corruption as well. Results for different $\sigma_{\min}$ values are given in Figure 4. Our improvement, although less dramatic than in the case of fog corruption, shows that decoupling the data-correction step from the training of the generative prior is a better approach for handling corruptions than [18] even for high-frequency corruptions.

Transport versus noise. Table 2 compares training on the transported data at all noise levels (transport only, $\sigma_{\min} = 0$) with Self-Cleaning Diffusion. Noise improves over transport alone for both corruptions (for fog, FID $11.12 \to 10.19$ and DINOv2 FD $371.91 \to 288.15$), and improves diversity on AFHQ (Table 3). The results are robust to $\sigma_{\min}$ (Figure 4): for blur, FID is nearly flat for $\sigma_{\min}$ between 0.186 and 0.670 (transported data used at 90% to 50% of the noise levels), and DINOv2 FD beats Ambient Omni for all $\sigma_{\min} \geq 0.397$. A sweep thus pays off, but $\sigma_{\min}$ need not be precise to beat Ambient Omni.

Table 2: Transport only ($\sigma_{\min} = 0$) versus transport + noise (90% fog and blur). Blur: the FID and DINOv2 FD optima of Figure 4.

	Method	$\sigma_{\min}$	FID ↓	DINOv2 FD ↓
Fog	Transport only	0.000	11.12	371.91
	Transport + noise (Self-Cleaning Diffusion)	0.397	**10.19**	**288.15**
Blur	Transport only	0.000	20.00	498.29
	Transport + noise (Self-Cleaning Diffusion)	0.397	**18.14**	432.69
		0.670	21.42	**393.88**

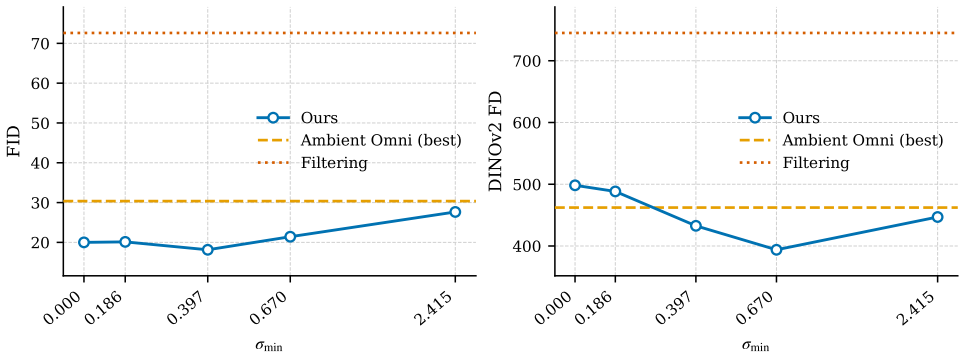

Figure 4: FID (left) and DINOv2 FD (right) as a function of $\sigma_{\min}$ for our method compared to the best result by Ambient Omni and the filtering approach in the 90% blur corruption setting. As seen, our method obtains better results than Ambient Omni even for high-frequency corruptions. The optimal value for FID seems to be $\sigma_{\min} = 0.397$, corresponding to using low-quality data for 70% of diffusion times, while that for DINOv2 FD seems to be $\sigma_{\min} = 0.670$, corresponding to using low-quality data for 50% of diffusion times.

4.2 Learning generative models by leveraging data from another class

In this section, we test our method for learning generative models using data that comes from a different class. We set up this experiment by using AFHQ [11] cats as the target distribution and AFHQ dogs as the out-of-distribution data. The question then becomes whether dog samples can help us train a generative model for cats.

The authors in [18] show that the noise level to merge these two classes is effectively infinity, and they only focus on extracting crops that can be used for high-frequency details during training. We hereby ask whether our transport map-based approach permits using out-of-distribution data to meaningfully learn low-frequencies from the target distribution. Without transport, the dogs can only be effectively used with a species label, which is the Conditional baseline of Table 3. Adding dogs without labels leads to the model generating dogs when we want cats, and using Ambient Omni leads to the same unless the added noise is infinite, *i.e.*, filtering out the dogs (Appendix E.3).

First, we train the transport map to go from dogs to cats, in the same way we go from corrupted to clean data. The transport map keeps common traits while substituting novel cat-like traits for the exclusively dog-like traits. Second, we train a diffusion model on original cats and the generated dogs-to-cat set with noise injected to avoid propagating transport errors. As seen in Table 3, our model gains meaningful diversity over both the filtering and class-conditional baselines while having equivalent class alignment and just slightly smaller aesthetic score [66]. Over 3 training seeds (Welch's t-test, Table 6), the Vendi gain over Conditional ($p = 0.017$) and the aesthetic gap ($p = 0.0006$) are significant, class alignment is not ($p = 0.80$). In Table 3, noise ($\sigma_{\min} = 0.094$) adds diversity over transport alone ($\sigma_{\min} = 0$).

4.3 Learning with poor aesthetics data

In this last experiment, we attempt to improve the aesthetic appearance of ImageNet while remaining faithful to its style. To this end, we split the ImageNet training set into high aesthetics, those with Aesthetic Score ≥ 5.0 [66], and low aesthetics, those with Aesthetic Score < 5.0. Next, we

Table 3: Performance comparison for the dogs and cats Out-of-Distribution learning experiment. Mean $\pm$ standard deviation over generation seeds.

Method	$\sigma_{\min}$	Class Alignment ↑	Aes. Score ↑	Vendi Score ↑
Filtering	∞	$\mathbf{0.9992 \pm 0.0001}$	4.893 ± 0.002	34.70 ± 0.06
Conditional	0.000	0.9867 ± 0.0004	$\mathbf{4.911 \pm 0.002}$	39.13 ± 0.03
Self-Cleaning Diffusion (Ours)	0.000	0.9953 ± 0.0002	4.812 ± 0.001	40.88 ± 0.17
	0.094	0.9912 ± 0.0001	4.805 ± 0.003	$\mathbf{41.49 \pm 0.02}$
	0.129	0.9923 ± 0.0002	4.766 ± 0.001	40.70 ± 0.04
	0.186	0.9898 ± 0.0004	4.732 ± 0.002	39.08 ± 0.11

Table 4: Performance comparison for the AestheticImageNet learning experiment.

Method	$\sigma_{\min}$	DINOv2 FD ↓	Aes. Score ↑	Aes. Score ≥ 5.0 ↑	Memorization ↓	Vendi Score ↑
Baseline	0.000	435	4.3443	8.39%	0.0728	**460.68**
Omni	0.186	478	4.4195	11.04%	**0.0624**	427.63
Self-Cleaning Diffusion (Ours)	0.186	**433**	**4.6411**	**20.86%**	0.1032	442.90

apply our method and first train a transport map from the low-aesthetic set to the high-aesthetic set, followed by noise injection and corruption-aware training [18] to obtain a generative model. Examples of transported and generated images can be found in Figure 5, where we observe how the transport map increases the visual quality of images while largely preserving their identity (left), leading to improved generation (right). Quantitatively, we see in Table 4 that our method improves average aesthetic score by 0.3 points, and more than doubles the percentage of high aesthetics (≥ 5) generations compared to the baseline. The improvements are a lot higher than with Ambient Omni. DINOv2 FD barely changes ($435 \to 433$) as it measures quality and diversity jointly: our method trades a little diversity (memorization, Vendi) for quality by overemphasizing high-quality samples during training, while the baseline treats all samples equally.

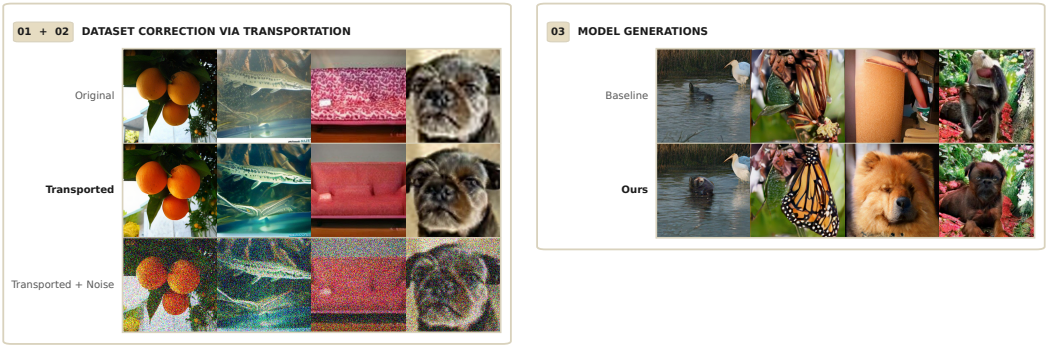

Figure 5: Visualization and comparisons of dataset correction (left) and generations by different approaches (right) in the aesthetic ImageNet setting. Transported images (left, row 2) exhibit visible artifacts, which we mask using noise (left, row 3), but are noticeably brighter, less clouded, better looking, and less noisy than the originals (left, row 1). Consequently, our approach obtains better average Aesthetic Score and a higher percentage of high aesthetics images than the models trained on the original ImageNet and with Ambient Omni (adding noise to the low aesthetics part of ImageNet).

4.4 Ablations of the transport stage

We evaluate the transport map by the median LPIPS [79] of the transported images to their clean versions (Appendix E.5). For fog, drifting is the best unpaired map (0.2915), ahead of unpaired rectified flow [47] (0.3374) and CycleGAN [82] (0.4011); the foggy inputs score 0.4981, the paired skyline 0.1073. The transport degrades gracefully with less clean data (LPIPS 0.192 with 10% and 0.214 with 5% clean data) and also corrects Gaussian noise, a color shift, and a weaker fog.

5 Conclusion

Self-Cleaning Diffusion is a principled technique for leveraging abundant low-quality and out-of-distribution data for generative model training. At a time when high-quality sample sources are close to exhausted, our work pushes the frontier of expanding the pool of useful and available data.

Acknowledgments and Disclosure of Funding

This research is supported by a Simons Investigator Award, a Simons Collaboration on Algorithmic Fairness, ONR MURI grants N00014-25-1-2116, N033697-00007, and N00014-26-1-2255, ONR grant N00014-25-1-2296, and support from Ericsson. Giannis Daras is supported by a Jane Street Research Collaboration award, a TPU Research Cloud grant, and compute grants from Lambda Research and Thinking Machines. The experiments were run on the Vista GPU Cluster through the Center for Generative AI (CGAI) and the Texas Advanced Computing Center (TACC) at UT Austin, and on the MIT Vision cluster.

Competing interests. The authors declare no competing interests.

References

[1] Asad Aali, Marius Arvinte, Sidharth Kumar, and Jonathan I Tamir. "Solving inverse problems with score-based generative priors learned from noisy data". In: *2023 57th Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2023, pp. 837–843 (cit. on pp. 2, 3).

[2] Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G Baraniuk. "Self-consuming generative models go mad". In: International Conference on Learning Representations (ICLR). 2024 (cit. on p. 2).

[3] Sina Alemohammad, Ahmed Imtiaz Humayun, Shruti Agarwal, John Collomosse, and Richard Baraniuk. "Self-improving diffusion models with synthetic data". In: *arXiv preprint arXiv:2408.16333* (2024) (cit. on pp. 2, 4, 21).

[4] Weimin Bai, Yifei Wang, Wenzheng Chen, and He Sun. "An expectation-maximization algorithm for training clean diffusion models from corrupted observations". In: *Advances in Neural Information Processing Systems* 37 (2024), pp. 19447–19471 (cit. on pp. 2, 3).

[5] Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. "Training diffusion models with reinforcement learning". In: *arXiv preprint arXiv:2305.13301* (2023) (cit. on pp. 4, 21).

[6] Ashish Bora, Eric Price, and Alexandros G. Dimakis. "AmbientGAN: Generative models from lossy measurements". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=Hy7fDog0b (cit. on p. 2).

[7] Mustapha Bounoua, Giulio Franzese, and Pietro Michiardi. *Learning to Match Unpaired Data with Minimum Entropy Coupling*. 2025. arXiv: 2503.08501 [cs.LG]. URL: https://arxiv.org/abs/2503.08501 (cit. on p. 6).

[8] Yann Brenier. "Polar factorization and monotone rearrangement of vector-valued functions". In: *Communications on pure and applied mathematics* 44.4 (1991), pp. 375–417 (cit. on p. 4).

[9] Junsong Chen, Jincheng Yu, Chongjian Ge, Lewei Yao, Enze Xie, Yue Wu, Zhongdao Wang, James Kwok, Ping Luo, Huchuan Lu, et al. "Pixart-a: Fast training of diffusion transformer for photorealistic text-to-image synthesis". In: *arXiv preprint arXiv:2310.00426* (2023) (cit. on pp. 2, 4, 20).

[10] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru R Zhang. "Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions". In: *arXiv preprint arXiv:2209.11215* (2022) (cit. on p. 3).

[11] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. "Stargan v2: Diverse image synthesis for multiple domains". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020, pp. 8188–8197 (cit. on pp. 9, 27).

[12] Xiaoliang Dai, Ji Hou, Chih-Yao Ma, Sam Tsai, Jialiang Wang, Rui Wang, Peizhao Zhang, Simon Vandenhende, Xiaofang Wang, Abhimanyu Dubey, et al. "Emu: Enhancing image generation models using photogenic needles in a haystack". In: *arXiv preprint arXiv:2309.15807* (2023) (cit. on pp. 2, 3, 20).

[13] Giannis Daras, Yuval Dagan, Alex Dimakis, and Constantinos Costis Daskalakis. "Consistent Diffusion Models: Mitigating Sampling Drift by Learning to be Consistent". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: https://openreview.net/forum?id=GfZGdJHj27 (cit. on p. 2).

[14] Giannis Daras, Mauricio Delbracio, Hossein Talebi, Alex Dimakis, and Peyman Milanfar. "Soft Diffusion: Score Matching with General Corruptions". In: *Transactions on Machine Learning Research* (2023). ISSN: 2835-8856. URL: https://openreview.net/forum?id=W98rebBxlQ (cit. on p. 3).

[15] Giannis Daras and Alex Dimakis. "Multiresolution Textual Inversion". In: *NeurIPS 2022 Workshop on Score-Based Methods*. 2022. URL: https://openreview.net/forum?id=3JCa_cqKaLy (cit. on p. 20).

[16] Giannis Daras, Alexandros G. Dimakis, and Constantinos Daskalakis. "Consistent diffusion meets tweedie: training exact ambient diffusion models with noisy data". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024 (cit. on pp. 2, 3).

[17] Giannis Daras, Jeffrey Ouyang-Zhang, Krithika Ravishankar, William Daspit, Costis Daskalakis, Qiang Liu, Adam Klivans, and Daniel J. Diaz. "Ambient Proteins: Training Diffusion Models on Low Quality Structures". In: *bioRxiv* (2025). DOI: 10.1101/2025.07.03.663105. eprint: https://www.biorxiv.org/content/early/2025/07/05/2025.07.03.663105.full.pdf. URL: https://www.biorxiv.org/content/early/2025/07/05/2025.07.03.663105 (cit. on pp. 2, 3, 31).

[18] Giannis Daras, Adrian Rodriguez-Munoz, Adam Klivans, Antonio Torralba, and Constantinos Daskalakis. "Ambient Diffusion Omni: Training Good Models with Bad Data". In: *arXiv preprint arXiv:2506.10038* (2025) (cit. on pp. 2–4, 7–10).

[19] Giannis Daras, Kulin Shah, Yuval Dagan, Aravind Gollakota, Alex Dimakis, and Adam Klivans. "Ambient Diffusion: Learning Clean Distributions from Corrupted Data". In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., 2023, pp. 288–313. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/012af729c5d14d279581fc8a5db975a1-Paper-Conference.pdf (cit. on pp. 2, 3).

[20] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848 (cit. on pp. 21, 27).

[21] Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. *Generative Modeling via Drifting*. en. Version Number: 2. 2026. DOI: 10.48550/ARXIV.2602.04770. URL: https://arxiv.org/abs/2602.04770 (visited on 04/15/2026) (cit. on pp. 4, 6, 21, 26–28).

[22] Prafulla Dhariwal and Alexander Nichol. "Diffusion models beat gans on image synthesis". In: *Advances in neural information processing systems* 34 (2021), pp. 8780–8794 (cit. on pp. 4, 21).

[23] Sander Dieleman. *Diffusion is spectral autoregression*. 2024. URL: https://sander.ai/2024/09/02/spectral-autoregression.html (cit. on p. 3).

[24] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. "Scaling rectified flow transformers for high-resolution image synthesis". In: *Forty-first international conference on machine learning*. 2024 (cit. on pp. 2, 3, 20).

[25] David J Field. "Relations between the statistics of natural images and the response properties of cortical cells". In: *Journal of the Optical Society of America A* 4.12 (1987), pp. 2379–2394 (cit. on p. 3).

[26] Dan Friedman and Adji Bousso Dieng. *The Vendi Score: A Diversity Evaluation Metric for Machine Learning*. 2023. arXiv: 2210.02410 [cs.LG]. URL: https://arxiv.org/abs/2210.02410 (cit. on p. 29).

[27] Samir Yitzhak Gadre et al. "DataComp: In search of the next generation of multimodal datasets". In: *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2023. URL: https://openreview.net/forum?id=dVaWCDMBof (cit. on pp. 2, 3, 20).

[28] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H Bermano, Gal Chechik, and Daniel Cohen-Or. "An image is worth one word: Personalizing text-to-image generation using textual inversion". In: *arXiv preprint arXiv:2208.01618* (2022) (cit. on p. 20).

[29] Matthias Gelbrich. "On a formula for the L^2 Wasserstein metric between measures on Euclidean and Hilbert spaces". In: *Mathematische Nachrichten* 147.1 (1990), pp. 185–203 (cit. on pp. 6, 19).

[30] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. "Scaling laws for autoregressive generative modeling". In: *arXiv preprint arXiv:2010.14701* (2020) (cit. on p. 2).

[31] 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*. 2018. arXiv: 1706.08500 [cs.LG]. URL: https://arxiv.org/abs/1706.08500 (cit. on p. 28).

[32] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS '20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546 (cit. on p. 2).

[33] Jonathan Ho and Tim Salimans. *Classifier-Free Diffusion Guidance*. 2022. arXiv: 2207.12598 [cs.LG]. URL: https://arxiv.org/abs/2207.12598 (cit. on pp. 4, 21, 26–28).

[34] Jordan Hoffmann et al. "Training compute-optimal large language models". In: *Proceedings of the 36th International Conference on Neural Information Processing Systems*. NIPS '22. New Orleans, LA, USA: Curran Associates Inc., 2022. ISBN: 9781713871088 (cit. on p. 2).

[35] Danial Hosseintabar, Fan Chen, Giannis Daras, Antonio Torralba, and Constantinos Daskalakis. "Diffem: Learning from corrupted data with diffusion models via expectation maximization". In: *arXiv preprint arXiv:2510.12691* (2025) (cit. on pp. 2, 3).

[36] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. "Lora: Low-rank adaptation of large language models." In: *Iclr* 1.2 (2022), p. 3 (cit. on p. 20).

[37] Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. "Generalization in diffusion models arises from geometry-adaptive harmonic representations". In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=ANvmVS2Yr0 (cit. on pp. 3, 6, 20).

[38] Mason Kamb and Surya Ganguli. "An analytic theory of creativity in convolutional diffusion models". In: *Forty-second International Conference on Machine Learning*. 2025. URL: https://openreview.net/forum?id=ilpL2qACla (cit. on pp. 6, 20).

[39] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. "Scaling laws for neural language models". In: *arXiv preprint arXiv:2001.08361* (2020) (cit. on p. 2).

[40] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. *Elucidating the Design Space of Diffusion-Based Generative Models*. 2022. arXiv: 2206.00364 [cs.CV]. URL: https://arxiv.org/abs/2206.00364 (cit. on pp. 2, 26, 27).

[41] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. *Analyzing and Improving the Training Dynamics of Diffusion Models*. arXiv:2312.02696 [cs]. Mar. 2024. DOI: 10.48550/arXiv.2312.02696. URL: http://arxiv.org/abs/2312.02696 (visited on 03/07/2026) (cit. on pp. 26, 27).

[42] Bahjat Kawar, Noam Elata, Tomer Michaeli, and Michael Elad. "Gsure-based diffusion model training with corrupted data". In: *arXiv preprint arXiv:2305.13128* (2023) (cit. on pp. 2, 3).

[43] Beomsu Kim, Gihyun Kwon, Kwanyoung Kim, and Jong Chul Ye. "Unpaired image-to-image translation via neural schr\" odinger bridge". In: *arXiv preprint arXiv:2305.15086* (2023) (cit. on p. 6).

[44] Jeffrey Li et al. "DataComp-LM: In search of the next generation of training sets for language models". In: *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*. 2024. URL: https://openreview.net/forum?id=CNWdWn47IE (cit. on pp. 2, 3, 20).

[45] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. "Flow matching for generative modeling". In: *arXiv preprint arXiv:2210.02747* (2022) (cit. on p. 6).

[46] Guan-Horng Liu, Arash Vahdat, De-An Huang, Evangelos A Theodorou, Weili Nie, and Anima Anandkumar. "I^2SB: Image-to-Image Schrödinger Bridge". In: *arXiv preprint arXiv:2302.05872* (2023) (cit. on p. 6).

[47] Xingchao Liu, Chengyue Gong, and Qiang Liu. "Flow straight and fast: Learning to generate and transfer data with rectified flow". In: *arXiv preprint arXiv:2209.03003* (2022) (cit. on pp. 2, 6, 10, 31).

[48] Haoye Lu, Qifan Wu, and Yaoliang Yu. "Stochastic Forward–Backward Deconvolution: Training Diffusion Models with Finite Noisy Datasets". In: *Forty-second International Conference on Machine Learning*. 2025. URL: https://openreview.net/forum?id=WrWqv3mpQx (cit. on pp. 2, 3).

[49] Haoye Lu, Yaoliang Yu, and Darren Ho. "SFBD-OMNI: Bridge models for lossy measurement restoration with limited clean samples". In: *arXiv preprint arXiv:2512.17051* (2025) (cit. on pp. 2, 3).

[50] Artem Lukoianov, Chenyang Yuan, Justin Solomon, and Vincent Sitzmann. "Locality in Image Diffusion Models Emerges from Data Statistics". In: *arXiv preprint arXiv:2509.09672* (2025) (cit. on pp. 6, 20).

[51] Yuta Matsuzaki, Seiichi Uchida, and Shumpei Takezaki. "SCoRe: Clean Image Generation from Diffusion Models Trained on Noisy Images". In: *arXiv preprint arXiv:2604.09436* (2026) (cit. on pp. 2, 3).

[52] Giacomo Meanti, Thomas Ryckeboer, Michael Arbel, and Julien Mairal. *Unsupervised Imaging Inverse Problems with Diffusion Distribution Matching*. 2025. arXiv: 2506.14605 [cs.CV]. URL: https://arxiv.org/abs/2506.14605 (cit. on p. 6).

[53] Chirag Modi, Jiequn Han, Eric Vanden-Eijnden, and Joan Bruna. "Generative Modeling from Black-box Corruptions via Self-Consistent Stochastic Interpolants". In: *arXiv preprint arXiv:2512.10857* (2025) (cit. on pp. 2, 3).

[54] Bernt Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013 (cit. on p. 3).

[55] Maxime Oquab et al. *DINOv2: Learning Robust Visual Features without Supervision*. 2024. arXiv: 2304.07193 [cs.CV]. URL: https://arxiv.org/abs/2304.07193 (cit. on pp. 28, 29).

[56] Vishakh Padmakumar and He He. "Does Writing with Language Models Reduce Content Diversity?" In: *The Twelfth International Conference on Learning Representations*. 2024. URL: https://openreview.net/forum?id=Feiz5HtCD0 (cit. on p. 2).

[57] Chicago Y Park, Shirin Shoushtari, Hongyu An, and Ulugbek S Kamilov. "Measurement score-based diffusion model". In: *arXiv preprint arXiv:2505.11853* (2025) (cit. on pp. 2, 3).

[58] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. "Sdxl: Improving latent diffusion models for high-resolution image synthesis". In: *arXiv preprint arXiv:2307.01952* (2023) (cit. on pp. 2–4, 20).

[59] 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. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV]. URL: https://arxiv.org/abs/2103.00020 (cit. on p. 29).

[60] Negin Raoof, Litu Rout, Giannis Daras, Sujay Sanghavi, Constantine Caramanis, Sanjay Shakkottai, and Alex Dimakis. "Infilling Score: A Pretraining Data Detection Algorithm for Large Language Models". In: *Statistical Foundations of LLMs and Foundation Models (NeurIPS 2024 Workshop)*. 2024. URL: https://openreview.net/forum?id=YovixNxQRv (cit. on pp. 6, 20).

[61] Adrián Rodríguez-Muñoz, William Daspit, Adam Klivans, Antonio Torralba, Constantinos Daskalakis, and Giannis Daras. "Ambient Dataloops: Generative Models for Dataset Refinement". In: *arXiv preprint arXiv:2601.15417* (2026) (cit. on pp. 2, 3).

[62] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-resolution image synthesis with latent diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 10684–10695 (cit. on p. 19).

[63] François Rozet, Gérôme Andry, Francois Lanusse, and Gilles Louppe. "Learning Diffusion Priors from Observations by Expectation Maximization". In: *The Thirty-eighth Annual Conference on Neural Information Processing Systems*. 2024. URL: https://openreview.net/forum?id=7v88Fh6iSM (cit. on pp. 2, 3).

[64] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. "Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 22500–22510 (cit. on p. 20).

[65] Tim Salimans and Jonathan Ho. "Progressive distillation for fast sampling of diffusion models". In: *arXiv preprint arXiv:2202.00512* (2022) (cit. on p. 2).

[66] Christoph Schuhmann. *Improved Aesthetic Predictor [GitHub]*. `https://github.com/christophschuhmann/improved-aesthetic-predictor`. 2022 (cit. on pp. 9, 29).

[67] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. "Laion-400m: Open dataset of clip-filtered 400 million image-text pairs". In: *arXiv preprint arXiv:2111.02114* (2021) (cit. on p. 2).

[68] Christoph Schuhmann et al. *LAION-5B: An open large-scale dataset for training next generation image-text models*. 2022. arXiv: `2210.08402 [cs.CV]`. URL: `https://arxiv.org/abs/2210.08402` (cit. on p. 29).

[69] Kulin Shah, Alkis Kalavasis, Adam Klivans, and Giannis Daras. "Does Generation Require Memorization? Creative Diffusion Models using Ambient Diffusion". In: *Forty-second International Conference on Machine Learning*. 2025. URL: `https://openreview.net/forum?id=GGPM0z3dhU` (cit. on pp. 2, 3, 6, 20).

[70] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. "Diffusion art or digital forgery? investigating data replication in diffusion models". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 6048–6058 (cit. on pp. 6, 20).

[71] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. "Understanding and Mitigating Copying in Diffusion Models". In: *Thirty-seventh Conference on Neural Information Processing Systems*. 2023. URL: `https://openreview.net/forum?id=HtMXRGbUMt` (cit. on pp. 6, 20).

[72] Yang Song and Stefano Ermon. "Generative Modeling by Estimating Gradients of the Data Distribution". In: *Advances in Neural Information Processing Systems*. 2019, pp. 11895–11907 (cit. on p. 2).

[73] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. "Score-Based Generative Modeling through Stochastic Differential Equations". In: *International Conference on Learning Representations*. 2021. URL: `https://openreview.net/forum?id=PxTIG12RRHS` (cit. on p. 3).

[74] George Stein, Jesse C. Cresswell, Rasa Hosseinzadeh, Yi Sui, Brendan Leigh Ross, Valentin Villecroze, Zhaoyan Liu, Anthony L. Caterini, J. Eric T. Taylor, and Gabriel Loaiza-Ganem. *Exposing flaws of generative model evaluation metrics and their unfair treatment of diffusion models*. 2023. arXiv: `2306.04675 [cs.LG]`. URL: `https://arxiv.org/abs/2306.04675` (cit. on p. 28).

[75] Antonio Torralba, Phillip Isola, and William T Freeman. *Foundations of computer vision*. MIT Press, 2024 (cit. on p. 3).

[76] Pablo Villalobos, Anson Ho, Jaime Sevilla, Tamay Besiroglu, Lennart Heim, and Marius Hobbhahn. "Position: will we run out of data? limits of LLM scaling based on human-generated data". In: *Proceedings of the 41st International Conference on Machine Learning*. ICML'24. Vienna, Austria: JMLR.org, 2024 (cit. on p. 2).

[77] Hanchen Wang, Tianfan Fu, Yuanqi Du, Wenhao Gao, Kexin Huang, Ziming Liu, Payal Chandak, Shengchao Liu, Peter Van Katwyk, Andreea Deac, et al. "Scientific discovery in the age of artificial intelligence". In: *Nature* 620.7972 (2023), pp. 47–60 (cit. on pp. 2, 31).

[78] Chen Henry Wu and Fernando De la Torre. "A latent space of stochastic diffusion models for zero-shot image editing and guidance". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 7378–7387 (cit. on p. 6).

[79] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. "The unreasonable effectiveness of deep features as a perceptual metric". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 586–595 (cit. on pp. 10, 30).

[80] Yinan Zhang, Eric Tzeng, Yilun Du, and Dmitry Kislyuk. "Large-scale reinforcement learning for diffusion models". In: *European Conference on Computer Vision*. Springer. 2024, pp. 1–17 (cit. on pp. 4, 21).

[81] Qingping Zheng, Yuanfan Guo, Jiankang Deng, Jianhua Han, Ying Li, Songcen Xu, and Hang Xu. "Any-size-diffusion: Toward efficient text-driven synthesis for any-size hd images". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 7. 2024, pp. 7571–7578 (cit. on pp. 2, 4, 20).

[82] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232 (cit. on pp. 6, 10, 31).

Table of Contents: Appendix

Contents

A Theoretical Analysis **18**

A.1 Omitted Proofs from Section 3.2 18

A.2 From W_2 Bounds to FID and DINOv2 FD 19

B Additional Background and Related Work **19**

B.1 Latent Space Diffusion 19

B.2 Variational Autoencoders 20

B.3 Controlling Overfitting to $\hat{p}_0$ 20

B.4 Natural Baselines 20

B.5 Drifting Models 21

C Training Details **21**

C.1 Controlled Experiments on ImageNet 21

C.2 Out-of-Distribution Data Experiments with AFHQ 27

C.3 Aesthetics Experiment on ImageNet 27

D Evaluation Metrics **28**

D.1 FID and DINOv2 FD 28

D.2 Memorization (sim > 0.8) 28

D.3 Vendi Score 29

D.4 Class (CLIP) Alignment 29

D.5 Aesthetic score 29

D.6 Error Bars 29

E Additional Results **29**

E.1 Selecting $\sigma_{\min}$ efficiently with an inference-based sweep 29

E.2 Why Transport Improves the FID of the Corrupted Data 30

E.3 Dogs and Cats without Transport 30

E.4 AFHQ: Noise Levels and Training Seeds 30

E.5 Transport-Map Ablations 30

F LLM Usage Declaration **31**

G Societal Impact **31**

H Limitations **31**

A Theoretical Analysis

A.1 Omitted Proofs from Section 3.2

Proof of Theorem 3.1. By the parallelogram identity applied to $a = x - \Phi(x)$, $b = T(x) - \Phi(x)$,

$$\left\|(1-s)\,x + s\,T(x) - \Phi(x)\right\|^2 \;=\; (1-s)\,\|x - \Phi(x)\|^2 \\ +\; s\,\|T(x) - \Phi(x)\|^2 \;-\; s(1-s)\,\|x - T(x)\|^2. \quad (7)$$

The expectations of the first and last terms on the RHS of the above with respect to $x \sim q_0$ are, respectively, $(1-s)\,W_2^2(q_0, p_0)$ and $-s(1-s)\,W_2^2(q_0, \hat{p}_0)$, because Φ and T are Brenier maps. To handle the middle term, expand $\|T(x) - \Phi(x)\|^2 = \|T(x) - x\|^2 + 2\langle T(x) - x,\, x - \Phi(x)\rangle + \|x - \Phi(x)\|^2$, whose expectation with respect to $x \sim q_0$ is $W_2^2(q_0, \hat{p}_0) + 2\beta + W_2^2(q_0, p_0)$. All in all, the expectation of (7) with respect to $x \sim q_0$ is:

$$W_2^2(q_0, p_0) + s^2 W_2^2(q_0, \hat{p}_0) + 2\beta s.$$

On the other hand, for $x \sim q_0$, the pair $((1-s)x + sT(x), \Phi(x))$ is a valid coupling of $\rho_s^\star$ and p_0. Therefore, the expectation of the LHS of (7) with respect to $x \sim q_0$ provides an upper bound to $W_2^2(\rho_s^\star, p_0)$. Putting the above together yields (4). □

Proof of Corollary 3.2. From Theorem 3.1, we have the upper bound:

$$W_2^2\big(\rho_s^\star,\, p_0\big) - W_2^2(q_0,\, p_0) \;\le\; s^2 W_2^2\big(q_0,\, \hat{p}_0\big) + 2s\beta$$

The first claim of the corollary is clear. If $\cos\theta > 0$, then $\beta < 0$ hence for small enough s the negative linear term on the right-hand side of the above overtakes the positive quadratic term.

Now let us move to the second claim. By definition, we substitute $\beta = -\cos\theta \cdot W_2(q_0, p_0) \cdot W_2(q_0, \hat{p}_0)$ into the inequality:

$$W_2^2\big(\rho_s^\star,\, p_0\big) - W_2^2(q_0,\, p_0) \;\le\; s^2 W_2^2(q_0, \hat{p}_0) - 2s\cos\theta W_2(q_0, p_0) W_2(q_0, \hat{p}_0).$$

We want to show that $W_2(\rho_s^\star, p_0) < W_2(q_0, p_0)$ for all $s \in (0, 1)$, which occurs if the right-hand side is strictly negative:

$$s^2 W_2^2(q_0, \hat{p}_0) - 2s\cos\theta W_2(q_0, p_0) W_2(q_0, \hat{p}_0) < 0$$

Since $s > 0$ and assuming $W_2(q_0, p_0) > 0$, we have:

$$\cos\theta > \frac{s}{2} \frac{W_2(q_0, \hat{p}_0)}{W_2(q_0, p_0)}$$

Because this must hold for all $s \in (0, 1)$, it is sufficient for the inequality to hold at the strict upper limit $s = 1$. Thus, if

$$\cos\theta > \frac{1}{2} \frac{W_2(q_0, \hat{p}_0)}{W_2(q_0, p_0)}$$

then the right-hand side is negative, and $W_2(\rho_s^\star, p_0) < W_2(q_0, p_0)$ for all $s \in (0, 1)$. □

Proof of Corollary 3.4. We define the upper bound of the distance difference as a function of the step size s:

$$f(s) = s^2 W_2^2(q_0, \hat{p}_0) - 2s\cos\theta W_2(q_0, p_0) W_2(q_0, \hat{p}_0).$$

This is a convex quadratic in s. The unconstrained minimizing s^* is

$$s^* = \cos\theta \frac{W_2(q_0, p_0)}{W_2(q_0, \hat{p}_0)}.$$

For s^* to be strictly within the interpolation interval $(0, 1)$ without overshooting, we require $s^* < 1$, which gives the necessary upper bound condition on $\cos\theta$:

$$\cos\theta < \frac{W_2(q_0, \hat{p}_0)}{W_2(q_0, p_0)}$$

Next, evaluating the quadratic $f(s)$ at $s = s^*$ gives the minimum value it can take, which gives the smallest possible $W_2^2\big(\rho^\star_{s^*},\, p_0\big)$ as follows:

$$W_2^2\big(\rho^\star_{s^*},\, p_0\big) - W_2^2(q_0,\, p_0) \;\leq\; (s^*)^2 W_2^2(q_0, \hat{p}_0) - 2s^* \cos\theta W_2(q_0, p_0) W_2(q_0, \hat{p}_0)$$

Substituting s^* back into this expression simplifies to:

$$W_2^2\big(\rho^\star_{s^*},\, p_0\big) - W_2^2(q_0,\, p_0) \;\leq\; -\cos^2\theta W_2^2(q_0, p_0)$$

Rearranging this, the best-case squared distance is bounded by:

$$W_2^2\big(\rho^\star_{s^*},\, p_0\big) \;\leq\; W_2^2(q_0, p_0) - \cos^2\theta W_2^2(q_0, p_0)$$

We require this optimally transported distance to be strictly less than $W_2^2(\hat{p}_0, p_0)$:

$$W_2^2(q_0, p_0) - \cos^2\theta W_2^2(q_0, p_0) < W_2^2(\hat{p}_0, p_0)$$

Rearranging to solve for $\cos\theta$:

$$\cos\theta > \frac{\sqrt{W_2^2(q_0, p_0) - W_2^2(\hat{p}_0, p_0)}}{W_2(q_0, p_0)}$$

Combining the lower bound (alignment requirement) and upper bound (no overshooting requirement) yields the stated condition, ensuring there exists an $s^* \in (0, 1)$ such that $W_2(\rho^\star_{s^*}, p_0) < W_2(\hat{p}_0, p_0)$. □

A.2 From W_2 Bounds to FID and DINOv2 FD

For a distribution μ on $\mathbb{R}^k$ with finite second moments, let $\mathcal{N}(\mu)$ be the Gaussian with the same mean and covariance. For a feature map $\phi : \mathbb{R}^d \to \mathbb{R}^k$ and $p, q \in \mathcal{P}_2(\mathbb{R}^d)$, the Fréchet distance of Appendix D is

$$\mathrm{FD}_\phi(p, q) = W_2^2\big(\mathcal{N}(\phi_\# p),\, \mathcal{N}(\phi_\# q)\big) = \|m_p - m_q\|^2 + \mathrm{Tr}\big(\Sigma_p + \Sigma_q - 2\,(\Sigma_p \Sigma_q)^{1/2}\big),$$

where m_p, Σ_p and m_q, Σ_q are the means and covariances of $\phi_\# p$ and $\phi_\# q$. FID uses the Inception-v3 features and DINOv2 FD the DINOv2 features.

Lemma A.1. *Let $p, q \in \mathcal{P}_2(\mathbb{R}^d)$ and let $\phi : \mathbb{R}^d \to \mathbb{R}^k$ be L-Lipschitz. Then*

$$\mathrm{FD}_\phi(p, q) \;\leq\; W_2^2(\phi_\# p, \phi_\# q) \;\leq\; L^2\, W_2^2(p, q).$$

The first inequality is Gelbrich's bound [29], and the second follows from the Lipschitz assumption. Combined with Theorem 3.1, this yields the desired upper bound.

B Additional Background and Related Work

B.1 Latent Space Diffusion

For high-dimensional distributions p_0, training a diffusion model is a computationally intensive process as both the input and the output of the network to be trained live in this high-dimensional space. In the seminal work of [62], the authors circumvent this issue by performing the diffusion in a latent space. In particular, the authors first train an encoder function $\mathrm{Enc} : \mathbb{R}^d \to \mathbb{R}^k$ and a decoder function $\mathrm{Dec} : \mathbb{R}^k \to \mathbb{R}^d$ that try to achieve lossless compression, i.e. these networks are trained such that:

$$\mathrm{Dec}\left(\mathrm{Enc}\left(X_0\right)\right) \approx X_0, \quad X_0 \sim p_0$$

while the dimension k remains small compared to d.

Given those functions, we can now train the diffusion model to target the lower-dimensional distribution induced by the encoder, i.e. we want to learn to sample random variables $Z_0 = \mathrm{Enc}(X_0), \quad X_0 \sim p_0$, since the decoder can map back the latents to the original space. The diffusion process in this latent space can be trained more efficiently because of the reduced dimensionality. In practice, the latent space for the diffusion modeling is obtained by training a Variational AutoEncoder (VAE). The VAE can be trained offline or jointly with the diffusion model.

B.2 Variational Autoencoders

To establish some notation, we use ω for the parameters of the encoder and ϕ for the parameters of the decoder. The encoder is typically chosen to be a stochastic function, parametrized as $\mathrm{Enc}_\omega(X_0) \sim \mathcal{N}(\mu_\omega(X_0), \sigma_\omega(X_0))$. We further write:

$$Z_0 = \mathrm{Enc}_\omega(X_0), \quad \bar{Z}_0 = \mu_\omega(X_0), \quad \hat{X}_0 = \mathrm{Dec}_\phi(Z_0), \quad \bar{X}_0 = \mathrm{Dec}_\phi(\bar{Z}_0).$$

The VAE is typically trained by minimizing the following objective:

$$J(\phi, \omega) = \mathbb{E}_{X_0, Z_0, \hat{X}_0}\Big[\underbrace{\|\hat{X}_0 - X_0\|_1 + \mathrm{LPIPS}(\hat{X}_0, X_0)}_{\text{reconstruction losses}} + \mathrm{d_{KL}}\big(\mathcal{N}(\mu_\omega(X_0), \sigma_\omega(X_0)), \mathcal{N}(0, I_d)\big)\Big].$$

In certain implementations, the VAE also has an additional adversarial loss where a discriminator is trained to distinguish between $\hat{X}_0$ and X_0. For the ease of presentation, we skip this adversarial loss for now.

B.3 Controlling Overfitting to $\hat{p}_0$

An important risk in our approach is that when we fit the transport map with limited samples from p_0, we can overfit to the empirical distribution $\hat{p}_0$ and not make meaningful progress towards the underlying continuous density. Indeed, generative models have been shown to memorize their training sets, especially when the sample size is small [37, 38, 50, 60, 69, 70, 71]. Our theoretical analysis controls this overfitting behavior via the s parameter in the regularized optimal transport problem of Eq. (3). In practice, we find that conditional models (like the ones we fit) collapse much less to the target distribution than their unconditional counterparts: we only observed overfitting to $\hat{p}_0$ when clean data was extremely scarce. We also note that while fully controlling the overfitting to the empirical as our theory requests is impossible, there are some practical ways to mitigate it by selecting appropriate model sizes, early stopping the training, or by augmenting the training set. That said, we did not find the need for such extensive regularizations. We provide all of our experimental parameters in Appendix Section C.

B.4 Natural Baselines

Despite the lack of theoretical guarantees, several empirical ideas have shown promise in the space of leveraging imperfect samples during the training of a diffusion model. We briefly list some below.

- **High-quality fine-tuning [12, 24, 58]**. A standard heuristic technique is to use all the data during the pre-training and then perform a finetuning stage in the in-distribution / high-quality set. Longer finetuning typically increases the quality of the generated samples at the cost of diversity. Practitioners control this trade-off by carefully tuning the finetuning duration or by constraining the number of parameters that can be changed in the finetuning stage [15, 28, 36, 64].
- **Data-mixing [24, 27, 44]**. An alternative approach is to sample from the out-of-distribution data sources more sporadically than the in-distribution data. This has a similar effect to reducing the strength of the gradient updates that come from the imperfect samples. Again, this approach requires careful tuning of the mixing ratio, and it often performs suboptimally compared to more sophisticated learning techniques.
- **Conditioning model on sample quality [9, 58, 81].** Another interesting approach is to explicitly condition the model during training with a label indicating the quality or the data source of the sample. At inference time, we can produce in-distribution samples by conditioning the model on the target label. The hope is that due to parameter sharing, the model will learn to leverage some of the out-of-distribution samples without introducing bias to the sampling distribution due to the explicit conditioning on the correct label.

We compare against some of those natural baselines in our Experiments Section 4. We also note that there are *inference-time* and *RL-based* methods for steering the distribution towards more desirable outcomes. Such techniques often trade coverage of the distribution for better sample quality. For the context of this paper, we primarily focus on training-time methods, and we underline that guidance [3, 22, 33] and RL techniques [5, 80] can be combined with our findings.

B.5 Drifting Models

An alternative to the diffusion paradigm was recently proposed by [21]. Instead of learning a sequence of estimators (as in diffusion), the authors of [21] propose learning an one-step generative model f_u that maps $\mathcal{N}(0, I_d)$ to the target distribution p_0.

In particular, for any two distributions p, q, the authors define a drifting field $V_{p,q}$ as follows:

$$V_{p,q}(x) = V_p^+(x) - V_q^-(x)$$

$$= \frac{1}{\mathbb{E}_{y^+\sim p}[k(x,y^+)]}\mathbb{E}_{y^+\sim p}\big[k(x,y^+)(y^+ - x)\big] - \frac{1}{\mathbb{E}_{y^-\sim q}[k(x,y^-)]}\mathbb{E}_{y^-\sim q}\big[k(x,y^-)(y^- - x)\big]$$

where $k(x, y)$ is a similarity kernel such as $\exp(-\tau||x - y||^2)$. Intuitively, the first term measures the expected similarity of x with samples from p and the second term the expected similarity of x with samples from q.

One observation is that if the two distributions p, q are equal, then the drifting field is 0. The authors of [21] use this insight to derive the following training objective:

$$J(u) = \mathbb{E}_{Z\sim\mathcal{N}(0,I_d)}\left[\left|\left|f_u(Z) - \text{sg}\left(f_u(Z) + V_{p_0,f_u\#\mathcal{N}(0,I_d)}(f_u(Z))\right)\right|\right|^2\right].$$

where sg is the stop-gradient operator. One minimizer of this objective is to find a function f_{u^*} that satisfies:

$$f_{u^*}\#\mathcal{N}(0, I_d) = p_0,$$

as desired. We underline that there are other (non-desired) solutions too, but empirically, the objective of Eq. (B.5) works well in practice.

We finally underline that Drifting Models can be generalized to map between any two distributions, i.e. the distribution $\mathcal{N}(0, I_d)$ can be replaced with any desired prior.

Algorithm 2 Pseudocode for the training of the drifting model.

```
# f: generator
# y_good: [N_pos, D], data samples from target distribution e.g. clean data
# y_bad: [N, D], data samples from prior distribution e.g. low quality data

x = f(y_bad) # [N, D], mapped samples from the prior distribution
y_neg = x # reuse x as negatives

V = compute_V(x, y_good, y_bad)
x_drifted = stopgrad(x + V)

loss = mse_loss(x - x_drifted)
```

C Training Details

C.1 Controlled Experiments on ImageNet

C.1.1 Dataset

ImageNet [20] consists of 1,281,167 images of variable resolution from 1000 classes. For this experiment, we resized all images to 256x256 using Lanczos downsampling (via PIL).

C.1.2 Corruptions

Fog corruption:

We used the following additive fog implementation with a severity of 5.

Algorithm 3 PyTorch Plasma-fractal Fog helpers

```
def _next_power_of_2(x: int) -> int:
    return 1 if x == 0 else 2 ** (x - 1).bit_length()

def _plasma_fractal(mapsize: int = 256, wibbledecay: float = 3.0) -> np.ndarray:
    """Diamond-square fog heightmap. Returns float32 array in [0, 1].

    Kept as NumPy because the algorithm is inherently sequential and only
    runs once per image (not per pixel), so there is no benefit to porting
    it to CUDA. Identical to the original ``corruptions.py`` implementation.
    """
    assert mapsize & (mapsize - 1) == 0, "mapsize must be a power of 2"
    maparray = np.empty((mapsize, mapsize), dtype=np.float32)
    maparray[0, 0] = 0
    stepsize = mapsize
    wibble = 100.0

    def wibbledmean(array: np.ndarray) -> np.ndarray:
        return array / 4 + wibble * np.random.uniform(-wibble, wibble, array.shape)

    def fillsquares() -> None:
        cornerref = maparray[0:mapsize:stepsize, 0:mapsize:stepsize]
        squareaccum = cornerref + np.roll(cornerref, shift=-1, axis=0)
        squareaccum += np.roll(squareaccum, shift=-1, axis=1)
        maparray[stepsize // 2:mapsize:stepsize,
                 stepsize // 2:mapsize:stepsize] = wibbledmean(squareaccum)

    def filldiamonds() -> None:
        ms = maparray.shape[0]
        drgrid = maparray[stepsize // 2:ms:stepsize, stepsize // 2:ms:stepsize]
        ulgrid = maparray[0:ms:stepsize, 0:ms:stepsize]
        ldrsum = drgrid + np.roll(drgrid, 1, axis=0)
        lulsum = ulgrid + np.roll(ulgrid, -1, axis=1)
        maparray[0:ms:stepsize,
                 stepsize // 2:ms:stepsize] = wibbledmean(ldrsum + lulsum)
        tdrsum = drgrid + np.roll(drgrid, 1, axis=1)
        tulsum = ulgrid + np.roll(ulgrid, -1, axis=0)
        maparray[stepsize // 2:ms:stepsize,
                 0:ms:stepsize] = wibbledmean(tdrsum + tulsum)

    while stepsize >= 2:
        fillsquares()
        filldiamonds()
        stepsize //= 2
        wibble /= wibbledecay

    maparray -= maparray.min()
    return maparray / maparray.max()
```

Algorithm 4 PyTorch Plasma-fractal Fog main

```
def plasma_fog(images: torch.Tensor, severity: int = 3) -> torch.Tensor:
    """Plasma-fractal fog corruption.

    A per-image fog layer is generated on the CPU with the diamond-square
    algorithm and added to the image on whatever device it resides on.

    severity 0 -> no fog; severity 9 -> near-total whiteout.
    """
    params = [
        (0.0, 2.0), # severity 0 - identity
        (1.5, 2.0),
        (2.0, 2.0),
        (2.5, 1.7),
        (2.5, 1.5),
        (3.0, 1.4),
        (3.5, 1.3),
        (4.0, 1.2),
        (4.5, 1.1),
        (5.0, 1.0),
    ]
    intensity, wibbledecay = params[severity]
    if intensity == 0.0:
        return images.clone()

    B, C, H, W = images.shape
    map_size = _next_power_of_2(max(H, W))

    # Work in [0, 1] to match the original's ``arr / 255.0`` arithmetic.
    arr = (images + 1.0) / 2.0 # [-1, 1] -> [0, 1]

    # Generate an independent fog layer per image.
    fog_layers = []
    for _ in range(B):
        fog_np = _plasma_fractal(mapsize=map_size, wibbledecay=wibbledecay)[:H, :W]
        fog_layers.append(torch.from_numpy(fog_np))
    # (B, H, W) -> (B, 1, H, W) for broadcasting over channels
    fog = torch.stack(fog_layers).unsqueeze(1).to(images.device, images.dtype)

    # Per-image max (matching the original's scalar ``arr.max()``).
    max_val = arr.view(B, -1).max(dim=1).values.view(B, 1, 1, 1)

    arr = arr + intensity * fog
    arr = arr * max_val / (max_val + intensity)

    return arr.clamp(0.0, 1.0) * 2.0 - 1.0 # [0, 1] -> [-1, 1]
```

And here are some examples of the corruption from the ImageNet validation set

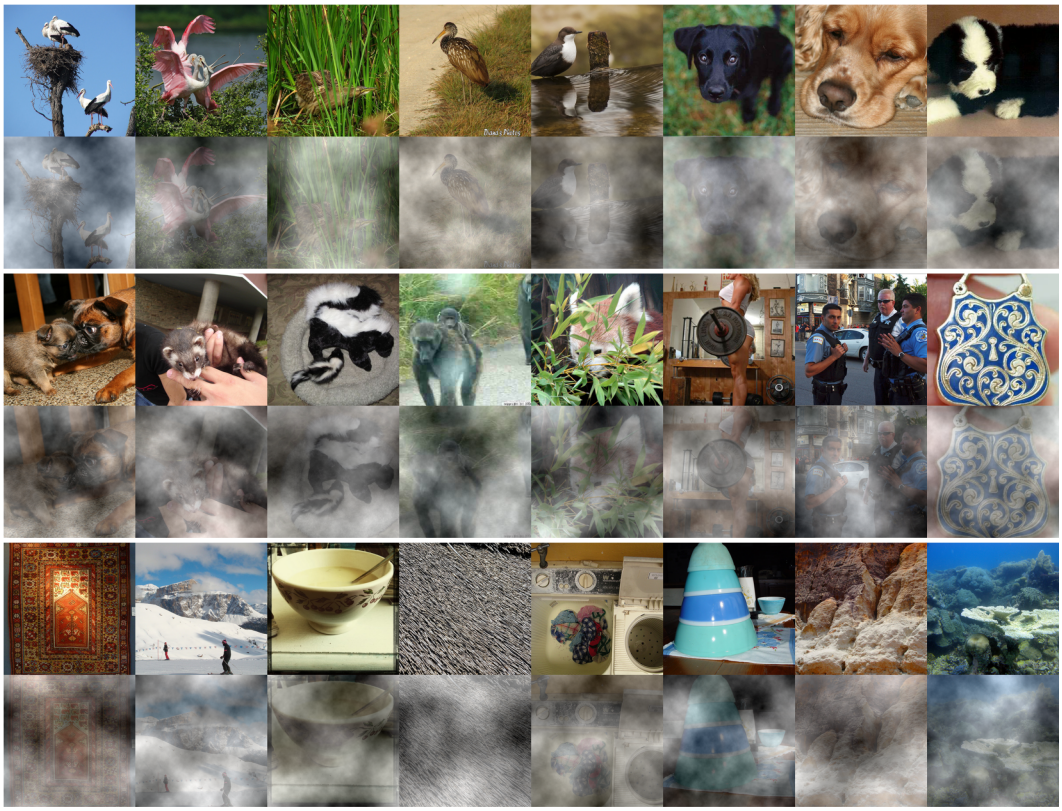

Figure 6: Examples of fog corrupted images

Blur corruption:

We used the following gaussian blur implementation with a severity of 3.

Algorithm 5 PyTorch Gaussian Blur

```
def _gaussian_kernel_1d(sigma: float, kernel_size: int) -> torch.Tensor:
    """Normalised 1-D Gaussian kernel."""
    x = torch.arange(kernel_size, dtype=torch.float64) - (kernel_size - 1) / 2.0
    kernel = torch.exp(-0.5 * (x / sigma) ** 2)
    return (kernel / kernel.sum()).float()

def gaussian_blur(images: torch.Tensor, severity: int = 3) -> torch.Tensor:
    """Gaussian blur matching ``PIL.ImageFilter.GaussianBlur(radius=...)``.

    Pillow uses ``kernel_size = 2 * radius + 1`` and internally derives
    ``sigma ~ 0.4 * radius + 0.6``. We reproduce both here with a
    separable depthwise convolution.
    """
    radius = [0, 1, 2, 3, 5, 7, 10, 14, 19, 25][severity]
    if radius == 0:
        return images.clone()

    kernel_size = 2 * radius + 1
    sigma = 0.4 * radius + 0.6

    k1d = _gaussian_kernel_1d(sigma, kernel_size)
    C = images.shape[1]
    # Separable: vertical then horizontal, applied per-channel.
    k_h = k1d.view(1, 1, kernel_size, 1).expand(C, 1, -1, -1).to(images.device,
        images.dtype)
    k_w = k1d.view(1, 1, 1, kernel_size).expand(C, 1, -1, -1).to(images.device,
        images.dtype)

    pad = radius
    out = F.pad(images, [0, 0, pad, pad], mode='reflect')
    out = F.conv2d(out, k_h, groups=C)
    out = F.pad(out, [pad, pad, 0, 0], mode='reflect')
    out = F.conv2d(out, k_w, groups=C)
    return out
```

And here are some examples of the corruption from the ImageNet validation set

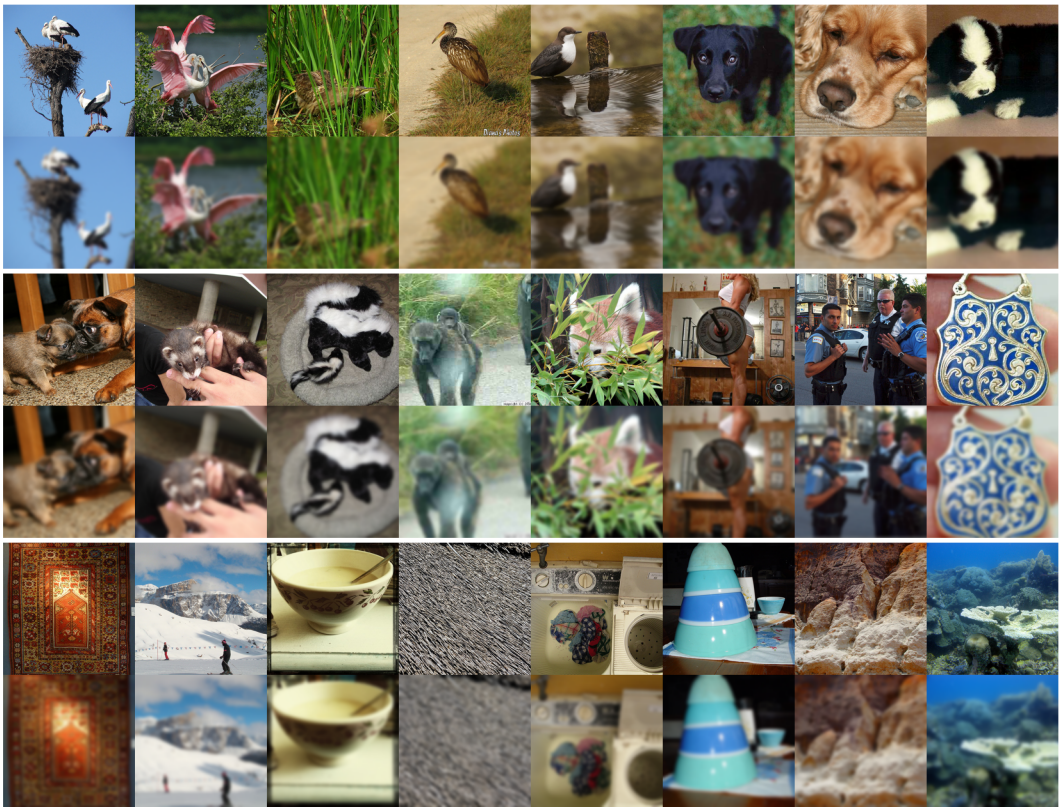

Figure 7: Examples of blur corrupted images

C.1.3 Diffusion Model

Architecture: We used the architecture of EDM2 [40, 41], specifically the XXL-512 configuration. It is a denoising U-Net with $\sim$ 1.4B parameters.

Training algorithm: We used a diffusion training algorithm with x0-prediction, the Variance Exploding (VE) formulation, and the EDM2 normalization. We trained all models unconditionally except the corruption-conditional model, which was trained conditioned on the corruption label. The models were trained for about $\sim$ 1B img with batch size 2048. The learning rate was 0.0065 using the Adam optimizer.

Inference algorithm: We used the EDM2 ODE sampler (Heun) with 32 steps. We also used the EDM2 model EMA weights with parameter 0.100. The corruption conditional model was evaluated using CFG guidance [33] 1.0 i.e. no guidance.

Compute and training time: Training the XXL-512 model took $\sim$ 4.5 days on 8xH100 GPUs.

C.1.4 Drifting Model

Architecture: We used the latents ablation configuration of the drifting paper [21]. It is a diffusion transformer with input size 32x32x4, patch size 2, hidden size 768, depth 12, num heads 12, mlp ratio 4.0, out channels 4, and approximately 133M parameters.

Training algorithm: We used the latent drifting algorithm with the latents ablation config. Specifically, we used 64 positives per sample, 16 negatives per sample (on top of the generated images), a positive bank size (per class) of 128, and a negative bank size of 1000. We trained for 30000 iterations. The learning rate was 0.0002 using the Adam optimizer, with 5000 warmup steps.

Inference algorithm: Drifting models are natively single step generators. We used the CFG algorithm in-built into the drifting model [33] with guidance scale 1.0, *i.e.*, no guidance, to transport the training data.

Compute and training time: Training the drifting model took $\sim$ 15 hours on 4xH200 GPUs.

C.2 Out-of-Distribution Data Experiments with AFHQ

C.2.1 Dataset

Animal Faces HQ (AFHQ) [11] consists of 16130 high-quality dog, cat, and wildlife images at 512×512 resolution. However, for our experiments we only use the 5653 cat images and 5239 dog images. We also resize them to 256x256 for computational reasons.

C.2.2 Diffusion Model

Architecture: We used the architecture of EDM2 [40, 41], specifically the XXS-512 configuration. It is a denoising U-Net with $\sim$ 30M parameters.

Training algorithm: We used a diffusion training algorithm with x0-prediction, the Variance Exploding (VE) formulation, and the EDM2 normalization. We trained all models unconditionally except the corruption-conditional model, which was trained conditioned on the corruption label. The models were trained for about $\sim$ 2B img with batch size 2048. The learning rate was 0.0170 using the Adam optimizer.

Inference algorithm: We used the EDM2 ODE sampler (Heun) with 32 steps. We also used the EDM2 model EMA weights with parameter 0.100. The corruption conditional model was evaluated using CFG guidance [33] 1.0 i.e. no guidance.

Compute and training time: Training the XXS-512 model took $\sim$ 3.5 days on 8xV100 GPUs.

C.2.3 Drifting Model

Architecture: We used the latents ablation configuration of the drifting paper [21]. It is a diffusion transformer with input size 32x32x4, patch size 2, hidden size 768, depth 12, num heads 12, mlp ratio 4.0, out channels 4, and approximately 132M parameters.

Training algorithm: We used the latent drifting algorithm with the latents ablation config adapted to a single-class setting. Specifically, we used 64 positives per sample, 0 additional negatives (just the generated images), a positive bank size of 1000, and no negative bank as no additional negatives were used. We trained for 30000 iterations. The learning rate was 0.0002 using the Adam optimizer, with 5000 warmup steps.

Inference algorithm: Drifting models are natively single step generators. We used no CFG, as this experiment had no classes (the prior and target distributions are each a single class).

Compute and training time: Training the drifting model took $\sim$ 15 hours on 4xH200 GPUs.

C.3 Aesthetics Experiment on ImageNet

C.3.1 Dataset

ImageNet [20] consists of 1,281,167 images of variable resolution from 1000 classes. For this experiment, we resized all images to 256x256 using Lanczos downsampling (via PIL).

C.3.2 Diffusion Model

Architecture: We used the architecture of EDM2 [40, 41], specifically the XS-512 configuration. It is a denoising U-Net with $\sim$ 118M parameters.

Training algorithm: We used a diffusion training algorithm with x0-prediction, the Variance Exploding (VE) formulation, and the EDM2 normalization. We trained all models unconditionally. The models were trained for about $\sim$ 2B img with batch size 2048. The learning rate was 0.0120 using the Adam optimizer.

Inference algorithm: We used the EDM2 ODE sampler (Heun) with 32 steps. We also used the EDM2 model EMA weights with parameter 0.100. The corruption conditional model was evaluated using CFG guidance [33] 1.0 i.e. no guidance.

Compute and training time: Training the XS-512 model took $\sim$ 2.2 days on 8xH100 GPUs.

C.3.3 Drifting Model

Architecture: We used the latents ablation configuration of the drifting paper [21]. It is a diffusion transformer with input size 32x32x4, patch size 2, hidden size 768, depth 12, num heads 12, mlp ratio 4.0, out channels 4, and approximately 133M parameters.

Training algorithm: We used the latent drifting algorithm with the latents ablation config. Specifically, we used 64 positives per sample, 16 negatives per sample (on top of the generated images), a positive bank size (per class) of 128, and a negative bank size of 1000. We trained for 40000 iterations. The learning rate was 0.0001 using the Adam optimizer, with 10000 warmup steps.

Inference algorithm: Drifting models are natively single step generators. We used the CFG algorithm in-built into the drifting model with guidance [33] 1.5.

Compute and training time: Training the drifting model took $\sim$ 15 hours on 4xH200 GPUs.

D Evaluation Metrics

D.1 FID and DINOv2 FD

Fréchet Inception Distance (FID) [31] and DINOv2 Fréchet Distance (DINOv2 FD) [55, 74] measure the difference between two distributions, in this case the generated and real image distributions. Features are extracted from both sets of images using a specific network (Inception-v3 or DINOv2 respecively), which are then modelled as a multivariate Gaussian by computing a mean and covariance matrix. The final distance is the Fréchet (Wasserstein-2) distance between the two feature multivariate Gaussians of the generated and real images

$$\text{FD} = \|\mu_r - \mu_g\|^2 + \text{Tr}\Big(\Sigma_r + \Sigma_g - 2\left(\Sigma_r\Sigma_g\right)^{1/2}\Big) \tag{8}$$

where μ_r, Σ_r and μ_g, Σ_g are the mean and covariance of the real and generated feature distributions, respectively.

Both FID and DINOv2 FD measure image quality / realistic-ness as well as diversity jointly within the same number. Thus, models which train on a small subset of the reference distribution, such as those obtained using the filtering method, or on a corrupted distribution, such as those obtained with too small a $\sigma_{\min}$, will obtain poor (high) FID and DINOv2 FD.

We measure FID and DINOv2 FD with respect to both the training set (labeled FID and DINOv2 FD in our tables) and the validation set (labeled as Val FID and Val DINOv2 FD in our tables). The first measures how well the model learned its own training distribution, while the second is a better measure of generalization as it uses a reference distribution of samples not seen during training.

D.2 Memorization (sim > 0.8)

Our memorization metric measures the fraction of generated samples with greater than 0.8 DINOv2 cosine similarity with some image in the training set.

$$\text{Mem}_{0.8} = \frac{1}{|\phi_g|}\sum_{x\in\phi_g} \mathbf{1}\left[\max_{y\in\phi_r} \text{cossim}(f(x),\, f(y)) > 0.8\right] \tag{9}$$

where $f(\cdot)$ denotes the DINOv2 embedding of an image. The selection of 0.8 as a threshold is heuristic, but visually we have seen that it generally works well.

D.3 Vendi Score

The Vendi Score (VS) [26] is a diversity metric defined as the exponential of the von Neumann entropy of the normalised similarity matrix given by some embedding. Given n samples and an embedding function $f : \mathcal{X} \to \mathbb{R}^d$ (e.g. DINOv2 [55]), we first calculate the cosine-similarity Gram matrix $K_{ij} = \mathrm{cossim}(f(x),\, f(y))$ and its normalisation $\hat{K} = K/\operatorname{tr}(K)$. The Vendi Score is then given by

$$\mathrm{VS}(X, f) = \exp\left(-\sum_{i=1}^{n} \lambda_i \log \lambda_i \right). \tag{10}$$

where $\lambda_1, \ldots, \lambda_n$ are the eigen values of $\hat{K}$. The Vendi Score ranges from 1 (all samples identical in embedding space) to n (maximally dissimilar), and can be roughly interpreted as the effective sample size.

D.4 Class (CLIP) Alignment

The Class Alignment metric uses CLIP [59] as a zero-shot classifier between two (or any number) of classes. For each class, we obtain an anchor text embedding by averaging CLIP text features of 28 prompt templates e.g., *"a photo of a {classname}"*, *"a dark photo of a {classname}"*, *"art of a {classname}"* where classname is something like "cat" or "dog", and then assign a probabiltiy to each image via a tempered softmax over the cosine similarities:

$$\mathrm{ClassAlign}(\phi_g) = \frac{1}{|\phi_g|} \sum_{x \in \phi_g} \frac{e^{\mathrm{cossim}(\mathrm{CLIP}_{\mathrm{img}}(x),\, \mathbf{t}_1)/\tau}}{e^{\mathrm{cossim}(\mathrm{CLIP}_{\mathrm{img}}(x),\, \mathbf{t}_1)/\tau} + e^{\mathrm{cossim}(\mathrm{CLIP}_{\mathrm{img}}(x),\, \mathbf{t}_2)/\tau}} \tag{11}$$

where ϕ_g is the generated image set, $\mathrm{CLIP}_{\mathrm{img}}(\cdot)$ is the CLIP image embedding, $\mathbf{t}_1, \mathbf{t}_2$ are the class anchor text embeddings, and $\tau = 1/e^s$ with s the learned CLIP logit scale. A score close to $\mathcal{A} = 1.0$ means the images are overwhelmingly of a specific class.

D.5 Aesthetic score

Aesthetic Score predicts the perceived visual quality of an image by passing its CLIP image embedding through a small MLP regressor trained on human aesthetic ratings [66, 68]. We report the score for a set ϕ_g of generated images as the mean score among images of the set:

$$\mathrm{AestheticScore}(\phi_g) = \frac{1}{|\phi_g|} \sum_{x \in \phi_g} \mathrm{MLP}(\mathrm{CLIP}_{\mathrm{img}}(x)) \tag{12}$$

Higher values indicate greater predicted aesthetic quality.

D.6 Error Bars

Training diffusion models at the ImageNet scale is too expensive for multiple training runs, so the error bars of Table 1 capture inference redundancy: FID and DINOv2 FD are computed on several sets of generated samples with different seeds, and we report their mean and standard deviation. For AFHQ (Table 3), we report the same inference redundancy for all models and, in Section 4.2, compare the two main models (Conditional and Self-Cleaning Diffusion with $\sigma_{\min} = 0.094$) over 3 training seeds with Welch's t-test.

E Additional Results

E.1 Selecting $\sigma_{\min}$ efficiently with an inference-based sweep

Table 5 compares the two-network sweep of Algorithm 1 with models trained at each $\sigma_{\min}$ (Figure 4) in the 90% blur setting. The sweep generates with a model trained on all the data at noise levels above σ_{th} and with its HQ fine-tune below. It selects the same $\sigma_{\min}$ for both metrics (0.397 for FID, 0.670 for DINOv2 FD); the Spearman rank correlation between the two columns is 0.7 for FID and

1.0 for DINOv2 FD. The sweep thus needs only the pretraining and HQ fine-tuning runs that standard pipelines already include, the generation of samples at each threshold, and one final fine-tuning run at the selected $\sigma_{\min}$, instead of one training run per candidate value.

Table 5: Two-network sweep versus training at each $\sigma_{\min}$, 90% blur.

	FID ↓		DINOv2 FD ↓	
$\sigma_{\min}$	**Two-network sweep**	**Trained**	**Two-network sweep**	**Trained**
0.000	26.25	20.00	561.92	498.29
0.186	23.27	20.13	492.96	488.36
0.397	**22.14**	**18.14**	455.65	432.69
0.670	23.43	21.42	**451.04**	**393.88**
2.415	27.96	27.65	456.53	446.96

E.2 Why Transport Improves the FID of the Corrupted Data

In the fog setting, the FID of the transported data is much lower than that of the foggy data (3.86 vs. 64.35, Figure 3), although the transported images show artifacts. A foggy image Y_0 retains most of the information of its clean version X_0, $H(X_0 \mid Y_0) \ll H(X_0)$, so it is easy for a model to recover the clean image from the foggy one. Yet the fog is a large perturbation in feature space: $\|f(X_0) - f(Y_0)\|^2$ is large for the Inception features f used by FID. Hence even an imperfect map, which removes the fog but introduces artifacts, improves FID drastically, and the noise of Stage 2 masks the remaining artifacts.

E.3 Dogs and Cats without Transport

Without a transport map, there are three ways to use the dogs of Section 4.2. (i) Mixing dogs and cats without noise gives a model that generates dogs in proportion to their share of the data, whereas the target is cats. (ii) Mixing them with noise, as in Ambient Omni, requires $\sigma_{\min} \to \infty$ for the dogs, because dogs and cats differ mainly in the low frequencies, for which Ambient Omni fails; this discards the dogs, *i.e.*, it reduces to filtering. (iii) Conditioning on the species label is the Conditional baseline, which is modestly worse than our method in Table 3 and much worse in the fog setting (Table 1).

E.4 AFHQ: Noise Levels and Training Seeds

In Table 3, $\sigma_{\min} = 0$ trains on the transported dogs at all noise levels (transport only), and $\sigma_{\min} = 0.094$, 0.129 and 0.186 use them for 97.5%, 95% and 90% of the diffusion training times. Of these values, $\sigma_{\min} = 0.094$ gives the highest diversity, both among the noise levels of Self-Cleaning Diffusion and compared with the baselines. Table 6 repeats the comparison of the two main models over 3 training seeds: the Vendi gain of Self-Cleaning Diffusion over Conditional and its lower aesthetic score are statistically significant, while the difference in class alignment is not.

Table 6: AFHQ over 3 training seeds: mean ± standard deviation and Welch's t-test.

Method	$\sigma_{\min}$	**Vendi Score ↑**	**Aes. Score ↑**	**Class Alignment ↑**
Self-Cleaning Diffusion (Ours)	0.094	$\mathbf{41.25 \pm 0.20}$	4.801 ± 0.007	0.9931 ± 0.0019
Conditional	0.000	38.32 ± 0.76	$\mathbf{4.919 \pm 0.012}$	0.9923 ± 0.0046
p-value (Welch, $n = 3$)	–	0.017	0.0006	0.80 (n.s.)

E.5 Transport-Map Ablations

Section 4.4 evaluates transport maps by their restoration score, the median LPIPS [79] between each transported image and its clean target (Tables 7–9). The maps are trained with 10% clean and 90% corrupted images (5% or 20% clean in Table 8).

- **Drifting:** the drifting transport map of our experiments.

- **Paired RF:** paired rectified flow (flow matching) [47], trained on (corrupted, clean) pairs, as a skyline reference.
- **Unpaired RF:** unpaired rectified flow (flow matching) [47].
- **CycleGAN** [82].

The choice of the transport map matters, with drifting models outperforming the other unpaired maps (Table 7). Still, any reasonable map improves over the two alternatives of training on the untransported samples (the foggy input) and discarding the OOD samples. Less clean data degrades the map, but it remains much better than the foggy input (Table 8). Across corruptions (Table 9), the weaker fog is easier to correct than the stronger one; surprisingly, the color shift has a slightly worse restoration score and FID than the weaker fog, despite a milder initial corruption (lower corruption score and FID), although the two are close.

Tables 8 and 9 use drifting transport maps trained with 5% or 20% clean data, or on data corrupted by fog of severity 3, additive Gaussian noise (standard deviation 0.30) or a color shift (HSV hue rotation by $180°$).

Table 7: Transport families, fog (severity 5). Median LPIPS of the transported images to their clean versions. Paired RF uses (corrupted, clean) pairs and serves as a skyline reference.

Map	**LPIPS** $\downarrow$
Corrupted input	0.4981
Paired RF (skyline)	0.1073
Drifting	0.2915
Unpaired RF	0.3374
CycleGAN	0.4011

Table 8: Fraction of clean data for the drifting transport, fog (severity 5).

Clean	**LPIPS** $\downarrow$
5%	0.214
10%	0.192
20%	0.182
Foggy input	0.498

Table 9: Drifting transport on other corruptions (10% clean): corrupted $\rightarrow$ transported.

Corruption	**LPIPS** $\downarrow$	**FID** $\downarrow$
Fog, severity 5	$0.498 \rightarrow 0.192$	$69.1 \rightarrow 4.29$
Fog, severity 3	$0.392 \rightarrow 0.148$	$26.4 \rightarrow 2.3$
Gaussian noise	$0.800 \rightarrow 0.277$	$74.5 \rightarrow 11.0$
Color shift	$0.319 \rightarrow 0.206$	$16.7 \rightarrow 2.7$

F LLM Usage Declaration

We used LLMs for coding, brainstorming, manuscript editing, rewriting, and assistance during the development of the theory but all the outputs were checked by the authors and there was a human in the loop. Specific usages were: Implementing standard methods, Proving theorems, Drafting sections of the paper, Editing (e.g., grammar, spelling, word choice), Visualizing results for submission, Facilitating or running experiments, Understanding technical concepts.

G Societal Impact

Given that (1) all the datasets we used are in the public domain and (2) prior works have already made public models trained on this data, we do not believe our work introduces extra risks that do not already exist. Positive impact of our work includes pushing the envelope of generative models in domains, like science, where collecting large-scale high-quality datasets is challenging [17, 77].

H Limitations

While the use of our method (transport map plus noise plus corruption-aware training) extracts more valuable knowledge from low-quality or out-of-distribution data than alternatives, it still has

limitations. Errors in the transport map may require a large amount of noise to mask, limiting benefits. Enough added noise always reduces to the filtering baseline.

There is also a memorization trade-off. A transport map trained with few target samples naturally decreases diversity by mapping the prior distribution to a narrower output which the model anchors on. Thus, our method memorizes more than the Conditional and HQ finetuning baselines. However, we also note that tuning $\sigma(t_{\text{min}})$ helps decrease memorization.

Our method also requires a key hyperparameter in $\sigma(t_{\text{min}})$. For best results, this should be decided via sweeping, although this is expensive computationally when every candidate value requires a training run, as in our experiments. The two-network sweep of Section 3.3 is much cheaper, and on blur it selects the same values as training (Appendix E.1). Our sweeps rank thresholds by FID and DINOv2 FD w.r.t. ground-truth statistics. When not available in practice, they may be substituted by a reward-based evaluation, as for other hyperparameters.

There is also the extra compute cost of training the drifting model on top of the diffusion model, which takes ~ 8 hours on 8×H200 or a $\sim 7\%$ overhead for the XXL models, and, with the two-network sweep, the generation of samples at each threshold and one more fine-tuning run.

Our theory analyzes the regularized optimal transport map of Eq. (3), whereas in practice we learn the map with a drifting model, which we treat as a practical, and possibly crude, approximation of it.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our claims accurately reflect the experiments and theoretical results obtained in the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See limitations Appendix Section H in the main paper.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: Our theoretical results provide the full set of assumptions and complete proof in Section 3.2 and Appendix Section A. The bound on FID and DINOv2 FD (Remark 3.6) additionally assumes a Lipschitz feature map, as stated in Lemma A.1.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide sufficient information about our method to program our algorithm in the Method section, the hyperparameters used in the main paper tables in the Experiments section, as well as the training configuration and computational requirements in the Appendix Section C. We also provide the implementation of the corruptions in the Appendix Section C.1.2. Lastly, our code is available at `https://github.com/adrianrm99/self-cleaning-release`.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility.

In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Our code is available at `https://github.com/adrianrm99/self-cleaning-release`. All data used is already publically available as for research use and was not created or publicized by us.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide training and test details including data and hyperparameters in Appendix Section C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Training several ImageNet-scale diffusion models per method is not feasible, so for the fog experiment we report inference-redundancy standard deviations of FID and DINOv2 FD (different generation seeds). For AFHQ, we report inference-redundancy standard deviations for all models and compare the two main models over 3 training seeds with Welch's t-test, which assumes normally distributed errors (Section 4.2). The captions and Appendix D.6 state what the error bars capture and how they are computed.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Appendix Section C Training Details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Our paper conforms to the NeurIPS code of ethics. All our experiments and results use publicly available data for which many models and extensive research has already been conducted.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: See Social Impact in Appendix Section G

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: All our experiments and results use publicly available data for which many models and extensive research has already been conducted. Thus, no risks are introduced by our paper. See Appendix Section G Societal Impact.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All data and models used are publically available for research purposes, and their authors are appropriately credited.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: No new assets introduced.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: See LLM usage declaration in Appendix Section F.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.