Learning a Maximum Entropy Model for Visual Textures using Diffusion

Xinyuan Zhao
New York University
xz2556@nyu.edu

Eero P. Simoncelli
New York University & Flatiron Institute
eero.simoncelli@nyu.edu

Abstract

Visual textures – spatially homogeneous image regions containing repeated elements (e.g. a field of grass, the bark of a tree) – are ubiquitous in visual scenes and provide important cues for recognizing and analyzing materials, surface geometry and objects. A number of existing texture models extract essential statistics from a single texture image, and can then generate high-quality samples that are visually similar to the original by matching these statistics. However, their statistics are either hand-designed or based on a network pretrained for another purpose (e.g., object recognition). Here, we develop a principled method for unsupervised learning of a set of statistics that are used to constrain a maximum entropy probability model for texture. We leverage methods developed for generative diffusion models to derive training and sampling procedures, and compare these to the traditional method of sampling via matching the statistics. Despite the compactness of our trained model (512 statistics), it generates texture images whose quality is as good as or better than the current state-of-the-art model ($\sim$177k statistics), both visually and by quantitative perceptual scoring using DreamSim. Finally, we show that images generated from parameters lying along a line between the parameter vectors corresponding to two different reference textures are homogeneous textures with features that smoothly transition between those of the two references.

1 Introduction

Most photographic images contain "visual textures" - homogeneous regions of repeated elements, subject to some randomization in location, size, color, orientation, or other attributes. Visual texture representation and synthesis methods have been extensively studied [e.g., Heeger and Bergen, 1995, De Bonet and Viola, 1997, Efros and Freeman, 2001, Zhu et al., 1998, Portilla and Simoncelli, 2000, Gatys et al., 2015, Lu et al., 2016, Ding et al., 2020, De Bortoli et al., 2021], and have served as a basis for style transfer [Gatys et al., 2016]. A bias toward texture (as opposed to shape) contributes to the misalignment between machine and human object recognition [Geirhos et al., 2018]. Visual textures also provide potent stimuli for visual neuroscience, where they have been used to differentiate the response selectivity of neurons in area V2 of the primate visual cortex relative to those in the preceding area V1 [Freeman et al., 2013, Yu et al., 2015].

Here, we aim to develop a *parametric* texture model, which takes a texture image as input and transforms it into a representation (a set of parameters or statistics), and in addition, a procedure for synthesizing new images of the same texture type based on this representation. Most existing parametric texture models are based on Julesz's seminal conjecture [Julesz, 1962]: the appearance of a texture is determined by a set of local statistics that are measured by the human visual system. Two textures with the same statistics will have the same (or similar) appearance. This concept has been instantiated in a variety of texture models and their corresponding sampling procedures (e.g., Heeger and Bergen [1995], Zhu et al. [1998], Portilla and Simoncelli [2000], Victor and Conte [2012], Gatys et al. [2015]). These models demonstrate their success by synthesizing images whose statistics match

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

those estimated from an original texture image, and showing that these are similar in appearance to the original. However, the statistics they use are either hand-designed or based on a pretrained object-recognition network.

The definition of a texture ensemble as a set of images with a given set of statistics can be formalized as a parametric probability model: images of a given texture class are samples drawn from the maximum entropy density [Jaynes, 1957] (i.e., the "most random" density) subject to consistency with the values of the chosen statistics [Zhu et al., 1998, Portilla and Simoncelli, 2000, Victor and Conte, 2012, Lu et al., 2016, De Bortoli et al., 2021]. In these previous instantiations, the statistics were hand-selected. Here, we develop a novel method for *unsupervised learning of a set of such statistics* from a dataset of texture images, and complementary methods for sampling from the maximum entropy density subject to these statistics. In this context, the vector of these statistics computed on any texture image provides a parameterization of the conditional density of the corresponding texture class.

To learn a maximum entropy conditional density model, we make use of "generative diffusion" methods that currently underlie most state-of-the-art image synthesis systems [Song and Ermon, 2019, Ho et al., 2020, Ramesh et al., 2022, Rombach et al., 2022]. The main concept is simple: a network trained to remove additive Gaussian noise from images provides an estimate of the score (gradient of the log probability) of the noisy image distribution. This gradient may then be used in an iterative reverse diffusion scheme to draw samples from a learned density of the training images. Diffusion models have been used to solve linear inverse problems using conditional sampling [Kadkhodaie and Simoncelli, 2021, Kawar et al., 2022, Chung et al., 2023, Daras et al., 2024]. The internal representation of denoisers used for diffusion has been shown to concentrate images with similar features in low-dimensional subspaces [Kadkhodaie et al., 2026]. More generally, they are often used to draw samples conditioned on auxiliary information such as class labels or text prompts. Diffusion models have also been used for the purpose of unsupervised representation learning (e.g., Fuest et al. [2026]). In many cases, the conditioning signal is provided by a separate (nonlinear) image representation [Mittal et al., 2023, Preechakul et al., 2022, Wang et al., 2023, Hudson et al., 2024, Yang and Mandt, 2023]. The conditioning and denoising networks are jointly trained on a set of images and their noisy counterparts. This configuration resembles an autoencoder, in which the representation network is the encoder, and the conditional denoiser is the decoder.

Here, we develop a model of similar structure – one network is used to compute a set of (learned) statistics, which are then used to condition a second network which performs denoising. But unlike previous methods, the architectural and algorithmic implementation of the conditioning is dictated by the mathematical form of the maximum entropy model that we assume. The primary contributions of the paper are:

- We develop a novel method for learning a set of statistics that can parameterize a maximum entropy density model for the training data. To our knowledge, this is the first of its kind.
- We train this model on a large ensemble of homogeneous image patches taken from ImageNet21K [Deng et al., 2009], and show on a test set that the conditional samples from the model closely resemble the texture images on which they are conditioned. The results are visually similar or superior to the current state-of-the-art parametric texture model [Gatys et al., 2015], while relying on a far smaller parameter set (512 vs. 176,640). We also achieve a better DreamSim score [Fu et al., 2023] for a majority of the homogeneous textures used for testing.
- We compare the conditional diffusion sampling procedure to the statistics matching procedure used to synthesize images in many previous texture models. Although these are mathematically closely related, they yield different results, with statistics-matching samples generally of higher quality.
- We examine the structure of the representation space by sampling from the model conditioned on parameter vectors interpolated between those of two different texture images. We find that the model of Gatys et al. [2015] generally produces images consisting of a patchwise spatial mixture of the two textures. In contrast, our model generates homogeneous texture samples whose features lie between the two textures.

The code for our model (including trained weights) is publicly available[1].

[1] `https://github.com/Bichidian/maxent-texture`

2 Method

2.1 Maximum entropy density model

We aim to learn a parametric family of probability densities over $x \in \mathbb{R}^n$ (for an n pixel image), where each density corresponds to a texture class, and individual texture images are samples. We assume a maximum entropy formulation [Jaynes, 1957], on which several existing texture models are based [Zhu et al., 1998, Portilla and Simoncelli, 2000, Lu et al., 2016, De Bortoli et al., 2021]. The maximum entropy probability distribution is the solution to the following optimization problem, parameterized by statistical values $\mu = [\mu_1, \mu_2, \cdots, \mu_d] \in \mathbb{R}^d$:

$$\max_p \mathbb{E}_p[-\log p(x)] \quad \text{s.t. } \mathbb{E}[f_k(x)] = \mu_k, \quad k = 1, 2, \cdots, d, \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}^d$ is a function that computes a vector of d statistics. The optimal solution is an "exponential family" (see Appendix A for the derivation):

$$p_\lambda(x) = \frac{1}{Z(\lambda)} \exp\left(-\lambda^T f(x)\right), \tag{2}$$

where $Z(\lambda)$ is a normalizing factor and $\lambda = [\lambda_1, \lambda_2, \cdots, \lambda_d] \in \mathbb{R}^d$ is chosen such that the density satisfies the constraints $\mathbb{E}_{p_\lambda}[f_k(x)] = \mu_k$. The two d-dimensional vectors λ and μ are uniquely determined by each other (Appendix A), and either can be used as a parametric specification of the conditional density of a texture class.

2.2 Training a maximum entropy model via denoising

Direct optimization of f and λ by maximizing likelihood over a training set is generally infeasible, since $Z(\lambda)$ requires integration over the entire data space. Instead, we propose to learn f indirectly via denoising, using a formulation similar to that of generative diffusion models. For a texture image x, consider the noise-contaminated y, expressed using the variance-preserving formulation of Ho et al. [2020]:

$$y = \frac{1}{\sqrt{\sigma^2 + 1}} x + \frac{\sigma}{\sqrt{\sigma^2 + 1}} \varepsilon, \tag{3}$$

where ε is a sample of Gaussian noise with zero mean and identity covariance. A function $\hat{\varepsilon}(y)$ trained to minimize the mean squared error in predicting ε will compute an approximation of the posterior mean, $\mathbb{E}(\varepsilon|y)$. This conditional mean, in turn, is proportional to the score (gradient of the log probability) of the noisy density $p(y)$ [Robbins, 1956, Miyasawa, 1961, Raphan and Simoncelli, 2011] (proof in Appendix B):

$$\mathbb{E}(\varepsilon|y) = -\frac{\sigma}{\sqrt{\sigma^2 + 1}} \nabla_y \log p(y). \tag{4}$$

For our texture model, we further assume that the noisy image y also follows the distribution given by (2), with a λ that depends both on the underlying texture class of x and on the noise level σ. This assumption is conceptually reasonable if we think that a noisy texture is still a texture, but the distribution in (2) is generally not closed under additive Gaussian noise, so it is best considered as an approximation. Given this, we can express the score of the maximum entropy density in (2) as:

$$\nabla_y \log p(y) = -\nabla_y [\lambda(x, \sigma)^T f(y)]. \tag{5}$$

Note that $Z(\lambda)$ disappears – one of the primary motivations for "score-matching" methods [Hyvärinen and Dayan, 2005]. We parameterize neural networks to compute approximations $f_\theta(y)$ and $\lambda_\phi(x, \sigma)$, and express our ε-predictor in terms of their inner product by combining (4) and (5):

$$\hat{\varepsilon}_{\theta,\phi}(y; x, \sigma) = \frac{\sigma}{\sqrt{\sigma^2 + 1}} \nabla_y [\lambda_\phi(x, \sigma)^T f_\theta(y)]. \tag{6}$$

Note that the dependency of $\hat{\varepsilon}$ on x is only through the low-dimensional vector $\lambda_\phi(x, \sigma)$ and does not make the denoising problem trivial.

In practice, we find that the model performs better when its output is trained to estimate a combination of signal and noise (so-called v-prediction [Salimans and Ho, 2022]):

$$v = \frac{1}{\sqrt{\sigma^2 + 1}} \varepsilon - \frac{\sigma}{\sqrt{\sigma^2 + 1}} x. \tag{7}$$

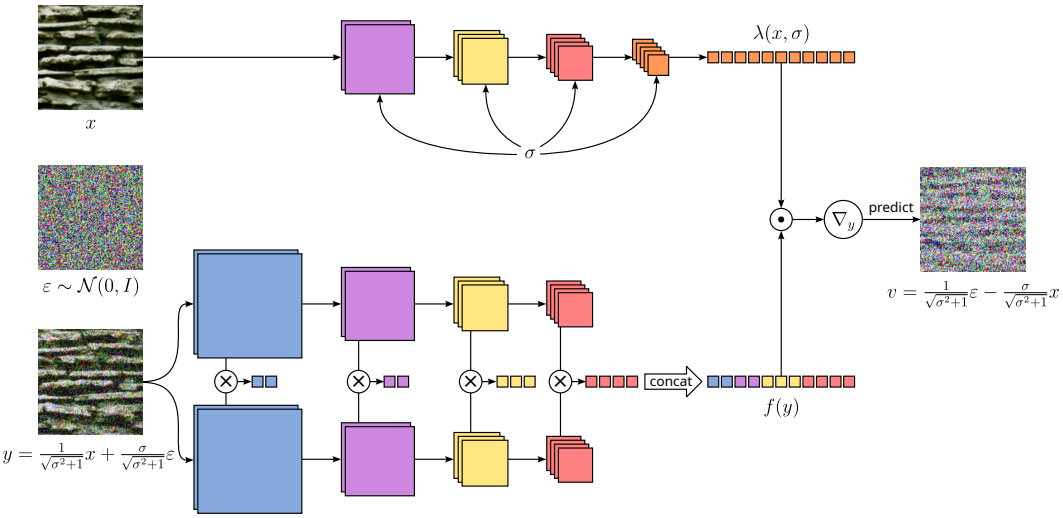


Figure 1: Model diagram. The top and bottom networks compute vectors $\lambda(x,\sigma)$ and $f(y)$, respectively. Input and output images $\{x, \varepsilon, y, v\}$ are 128x128 in resolution. Blue, purple, yellow, red, orange colors represent channels with spatial resolutions 128x128, 64x64, 32x32, 16x16, 8x8, respectively (not drawn to scale). Top network is based on a ConvNeXt-T model [Liu et al., 2022], and the $\lambda(x,\sigma)$ vector consists of spatial averages of channels at the last stage of the network. Bottom network is constructed from a twin pair of UNet encoders [Rombach et al., 2022], initialized and trained independently. The $f(y)$ vector consists of the inner products of corresponding channels in the two UNet encoders (for the final layer of each scale, just before downsampling). The gradient of the inner product of the two vectors provides an estimate of v (see text).

Now the effective ε-predictor becomes:

$$\hat{\varepsilon}_{\theta,\phi}(y;x,\sigma) = \frac{\sigma}{\sqrt{\sigma^2+1}}y - \frac{1}{\sqrt{\sigma^2+1}}\nabla_y[\lambda_\phi(x,\sigma)^T f_\theta(y)]. \tag{8}$$

This reparameterization from ε-prediction to v-prediction is without loss of generality if the span of $f(y)$ includes the squared norm $y^T y$ (i.e. there exists a λ such that $\nabla_y[\lambda^T f(y)] = y$), since the difference can be absorbed into λ. With this form of $\hat{\varepsilon}_{\theta,\phi}(y;x,\sigma)$, we optimize:

$$\min_{\theta,\phi} \mathbb{E}_{x,\sigma,\varepsilon} \left\| \hat{\varepsilon}_{\theta,\phi}(y;x,\sigma) - \varepsilon \right\|^2. \tag{9}$$

This is a minimum mean squared error (MMSE) denoising objective (or equivalently, ε-predictor). When optimized over a wide range of noise levels σ, it provides a substrate for a generative diffusion model.

2.3 Dataset

There are several existing texture datasets (e.g., DTD [Cimpoi et al., 2014], ALOT [Burghouts and Geusebroek, 2009], OpenSurfaces [Bell et al., 2013]), but these are generally not large enough to train an expressive neural network from scratch (DTD: 5640 images of 47 classes; ALOT: ∼27500 images of 250 classes; OpenSurfaces: 27796 texture regions contained in bigger images). In addition, many images in these dataset are substantially spatially inhomogeneous in luminance, color, orientation, scale and other features, due to perspective, lighting and inherently inhomogeneous structure (e.g. a single large vein on a leaf), but our model (as most texture models) assumes spatial homogeneity. Significantly larger datasets all consist of generic (mostly inhomogeneous) natural images, so we need to select a homogeneous subset for training purposes.

To this end, we crop the natural images in the ImageNet21K dataset [Deng et al., 2009] into non-overlapping 128x128 images patches, and then select the most "homogeneous" patches using a simple criterion. We first decompose each patch using the steerable pyramid [Simoncelli and Freeman, 1995] (implemented by Duong et al. [2023]), a linear transform that partitions the image into feature maps

at different orientations and scales. We use a pyramid with 3 scales and 4 orientations, resulting in 4 oriented feature maps with 64x64, 32x32 and 16x16 resolutions, as well as a 128x128 map of the high-pass residual and a 16x16 map of the low-pass residual. We square all 14 feature maps (except for the low-pass residual), and recursively blur and downsample them to 16x16. For each, we compute a homogeneity score - the standard deviation divided by the mean - and sum these over all maps. We chose the one million homogeneous patches with lowest scores as suitable for training our network (see examples in Figure 6 in the Appendix).

We do not need a large number of texture images for testing, so we use the Describable Textures Dataset (DTD) [Cimpoi et al., 2014] as the test set. This avoids overlap with the training images (patches drawn from ImageNet), and provides a convenient benchmark on a public database for future comparisons. For preprocessing, we resize each DTD image so that its shorter side has length 128, and then center crop it to 128x128. We use all 5640 DTD images for quantitative evaluations. For visualization, we select 12 images based on the same homogeneity score that we used to generate the training set. Specifically, we compute homogeneity scores for all DTD images, and find that 599 (out of 5640) are above the threshold for training images. We divide these into 12 subsets (each containing 50) and select one image from each subset to ensure the quality and diversity among the final 12 images that are shown in Figures 2 and 9. We also choose 6 inhomogeneous images (which we do not consider as proper textures) to demonstrate the out-of-distribution behavior of the model (Figure 10).

2.4 Network architecture and training

To construct $\lambda_\phi(x, \sigma)$, we use a ConvNeXt-T model [Liu et al., 2022] and introduce dependency on σ via a sinusoidal embedding and per-channel FiLM modulation [Perez et al., 2018] at the end of each convolution block. We replace the stride of the stem convolution from 4 to 2, so that the network can operate on a 128x128 input. See Figure 1 for a diagram.

To construct $f_\theta(y)$, we take the first half of a UNet architecture [Rombach et al., 2022] and treat the channel activation at the end of each scale (those that would enter the second half of the UNet as skip connections) as the network output. We use two such networks of identical architecture, but whose parameters are initialized and trained independently. We feed them with the same noisy image y and compute inner products between their corresponding output channels, producing one scalar value per channel. The network output, $f_\theta(y)$, corresponds to the vector containing these scalars.

Here we provide justification for this non-standard "twin half-UNet" architecture of the f network. Our initial attempt was to use an identical ConvNeXt as the λ network (without σ conditioning), but this completely failed on texture generation. We reasoned that we needed to build a network whose *gradient* resembles a UNet, which is known to perform well in denoising. Specifically, the network needs to contain input-to-output paths on different spatial scales, such that features of any spatial scale can be computed without going through coarser scales (in a UNet this is enabled by the skip connections). To satisfy this, we tried using the first half of a UNet architecture and computing output by taking the spatial average of each scale, which resulted in a dramatic improvement. We further improved performance by using two identical half-UNet architectures and taking the inner product between corresponding channels along the spatial dimensions.

We set the number of statistics to $d = 512$ (increasing this to 1024 did not noticeably increase the performance). We choose a log-normal training distribution for σ: $\log \sigma \sim \mathcal{N}(0.0, 0.6^2)$. This follows Karras et al. [2022], except that the mean is larger and the variance is smaller, which we find to work better for our purpose. This distribution is also close to the "inverse schedule" used in Hudson et al. [2024]. We use our curated texture dataset (described in 2.3), normalize the images to the range [-1, 1], and use random crop augmentation. We train our network for 10 epochs with batch size 64, using Adam optimizer [Kingma and Ba, 2014] in PyTorch [Ansel et al., 2024]. The training takes 25 hours on four NVIDIA A100 GPUs.

2.5 Image representation and generative sampling

Under the maximum entropy framework, both $\lambda(x, \sigma)$ and $f(x)$ can be considered as the "representation" of a texture image x. Note that given x, the former still explicitly depends on σ. Also note that although the function f takes the noisy image y as its input during training, here we treat it as a representation model for the clean image x. Since $f(x)$ is computed on a homogeneous image as a spatial average of convolution channels, the value of $f(x)$ is almost always very close to its expectation μ,

by the law of large numbers (more formally, by the equivalence of ensembles or concentration of measure [Talagrand, 1996]). Therefore, we can assume the approximation $\mu \approx f(x)$. Furthermore, since μ and λ are uniquely determined by each other in the maximum entropy formalism (Appendix A), the two representations $\lambda(x, \sigma)$ and $f(x)$ are approximately equivalent. We can generate new texture images from either of these two representations, by using one of the following two methods. Appendix C explains how these two methods are mathematically related.

Statistics matching Given a reference texture x_0, we measure its statistics $\mu_0 = f(x_0)$ and solve the following optimization problem:

$$\min_x \|f(x) - \mu_0\|^2 \tag{10}$$

We initialize x with i.i.d uniform distribution in the allowed range [-1, 1], and optimize with Adam optimizer for 5000 steps. This method is used in a number of existing texture models, including Gatys et al. [2015], against which we compare our results.

Diffusion sampling Given a schedule of σ_t and the corresponding $\lambda_t = \lambda(x_0, \sigma_t)$ for a reference texture x_0, we perform the standard DDPM reverse diffusion procedure [Ho et al., 2020]:

$$x_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(x_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \hat{\varepsilon}(x_t; \lambda_t) \right) + \sqrt{\frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}(1 - \alpha_t)}\eta, \tag{11}$$

where $\bar{\alpha}_t = 1/(1 + \sigma_t^2)$ and $\alpha_t = \bar{\alpha}_t / \bar{\alpha}_{t-1}$, and η is an identity-covariance Gaussian noise. We use a schedule in which σ_t goes from 10.0 to 0.1 in 1000 steps, uniformly spaced on a log scale.

2.6 Baseline models

We compare textures with our model to those of several existing *parametric* texture models. There are other methods for texture synthesis, such as text-based methods [Wang et al., 2024, Guerrero-Viu et al., 2024], but these are not directly comparable to our model.

Portilla-Simoncelli model Portilla and Simoncelli [2000] developed a parametric texture model inspired by biological vision, based on a set of hand-selected statistics measured in a multi-scale image representation (a "steerable pyramid") [Simoncelli and Freeman, 1995]. Samples were drawn by matching statistics using a custom gradient-based optimization procedure. The original model only considered grayscale images, but a publicly available extensions provides an extension to color images[2], which we use for comparison. The color model is based on 6,495 statistics, but about half of these are redundant (due to the symmetry of covariance matrices), yielding $\sim$3k unique statistics.

μNCA model Mordvintsev and Niklasson [2021] proposed a texture model based on a cellular automata with a small number of free parameters. For each given texture, these parameters need to be trained so that the synthesized image after several iterations is close to the original, according to their hand-designed metric. We use the best performing version of their model, which has 588 parameters, and follow their official implementation[3].

Gatys model The model introduced in Gatys et al. [2015] is currently considered the state-of-the-art for parametric texture representation and generation. The statistics it uses are the spatial mean and cross-channel covariance of the hidden layers of the VGG19 network [Simonyan and Zisserman, 2014] that is pretrained on ImageNet classification. Since the dimensionality of the covariance is proportional to the square of the number of channels in a hidden layer, the Gatys model consists of a very large number of statistics (the standard version has 176,640). New textures are generated via statistics matching. We find that high-quality synthesis requires initialization of x with a small noise amplitude (we use a uniform distribution over 1% of the allowed range) since the model otherwise produces high-frequency noisy artifacts in the generated texture images (see D.1).

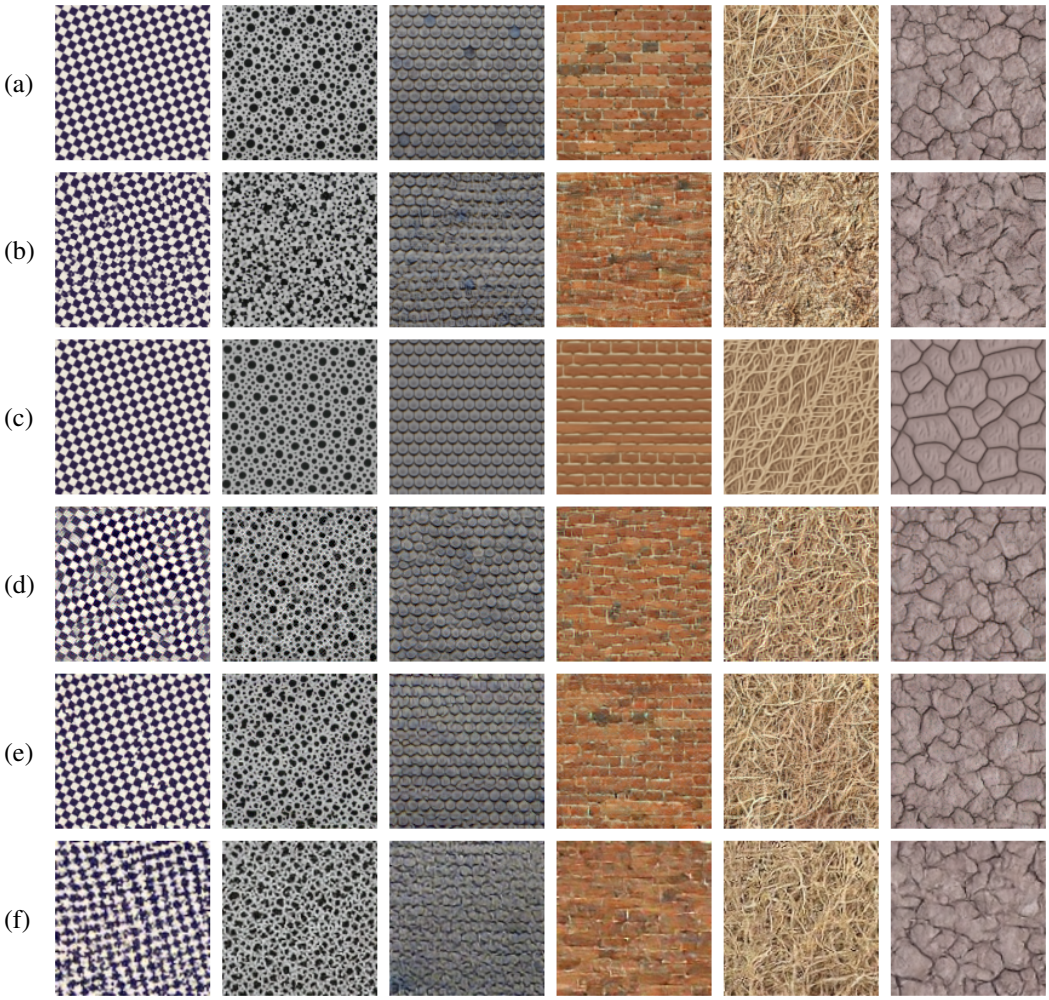


Figure 2: Resampling examples. **(a)** Original texture images (from DTD dataset, sorted in decreasing homogeneity). Remaining columns show samples generated by **(b)** Portilla-Simoncelli model; **(c)** μNCA model; **(d)** Gatys model; **(e)** statistics matching for our model; **(f)** diffusion sampling for our model.

3 Results

3.1 Texture resampling

Given an original texture image x_0, we can synthesize a new sample (i.e. generate a new texture image in the same class as x_0) by matching statistics $\mu_0 = f(x_0)$, or by diffusion sampling conditioned on $\lambda_0 = \lambda(x_0, \sigma)$. Over a set of test images, we find that statistics matching generates better quality images than diffusion sampling (Figure 2; additional examples in Figure 9; examples for inhomogeneous images in Figure 10). For comparison, we find that our model (with statistics matching) generates texture images whose visual quality is similar to or better than those generated with the Gatys model, and is significantly better than the Portilla-Simoncelli model. The μNCA model performs very well on textures that are largely periodic, but produces overly smooth or stylized samples for less regular textures. Note that our model uses significantly fewer (512) parameters than the other models, especially than the Gatys model (176,640 parameters, which exceeds the number of pixels in the 128x128 training images). It is intriguing that the two sampling methods for our model lead to different resampling quality, since the two representations μ and λ are related by a one-to-one

[2] `https://www.cns.nyu.edu/~lcv/texture/`

[3] `https://github.com/google-research/self-organising-systems`

Table 1: Comparison of our model (with statistics matching) and Gatys model using DreamSim score. For each original image, Welch's t-test is performed using 10 samples for each model, and $p > 0.05$ is deemed a tie.

	Total	Ours better	Gatys better	Tie
All DTD images	5640	2398	2112	1130
More homogeneous	599	411	118	70
Less homogeneous	5041	1987	1994	1060

correspondence (bijection). In addition, as we explain in Appendix C, the two sampling methods are also mathematically related.

3.2 Perceptual quality metric

To quantitatively compare the sample quality of different models, we compute the DreamSim score [Fu et al., 2023] between original and resampled images. We use all 5,640 images in the DTD dataset [Cimpoi et al., 2014] as originals, and generate 10 resampled images per original, for our model (with statistics sampling) and the Gatys model. We find that scores vary substantially across the images in the DTD dataset, and are dependent on the homogeneity score of the original (as defined in 2.3). Out of the 5,640 images, 599 images are more homogeneous than the threshold that we used for the selection of the training set (which resulted in 1 million training patches). The rest (5,041 images) are less homogeneous than the threshold. As the left panel of Figure 3 shows, both models perform better (have smaller scores) on the more homogeneous images (orange dots) than the less homogeneous images (blue dots). Importantly, the two models perform similarly on the less homogeneous images, but our model (with statistical matching) outperforms the Gatys model on a majority of the more homogeneous images (Table 1). The right panel of Figure 3 shows that for our model, statistics matching outperforms diffusion sampling on almost all DTD images.

The DreamSim score (like other perceptual quality scores such as DISTS [Ding et al., 2020] and LPIPS [Zhang et al., 2018]) is computed on pairs of individual images (one original and one sample), and therefore cannot quantify sample diversity. Visually, when we repeat the sampling procedures with different random seeds, all methods seem to generate different samples (Figures 7 and 8 in the Appendix). Although this shows that the models do not always generate the same image (which could happen if the model memorizes the training set), it does not reveal more subtle aspects of diversity. To quantify diversity, we computed the FID scores (Appendix D.2), which estimates a distance between the distributions of latent representations of true and generated images, providing a measure of both quality and diversity.

3.3 Interpolation

We examine the structure and generalization capabilities of our model's representation space by generating texture samples whose parameters lie between those of two existing textures x_a and x_b. Linear interpolation can be performed in μ or λ space, using statistics matching or diffusion sampling, respectively:

$$\mu_m = \frac{M-m}{M} f(x_a) + \frac{m}{M} f(x_b), \quad \text{or} \quad \lambda_m(\sigma) = \frac{M-m}{M} \lambda(x_a, \sigma) + \frac{m}{M} \lambda(x_b, \sigma), \qquad m = 0, \cdots, M \tag{12}$$

Note that interpolation in these two spaces is generally not equivalent: λ and μ are related by a highly nonlinear bijection, so a straight line in one space is likely to be curved in the other. For comparison, we perform the same task using other parametric texture models (Portilla-Simoncelli, μNCA, Gatys) by linearly interpolating (going in straight lines) in their respective representation spaces.

Figure 4 provides a set of example interpolations (additional examples are provided in Appendix Figures 12 and 13). Interpolation results for μNCA are not shown because the cellular automata dynamics becomes unstable with the interpolated parameters. Interpolation in the pixel domain (i.e., "fading" between images) produces "double exposure" images [row (a)]. Samples generated to match interpolated statistics in the Gatys model [row (c)] are patchwise spatial mixtures of the two endpoint textures, rather than homogeneous textures with features that lie between those of the endpoint textures. If we consider the valid support set of these models to be restricted to homogeneous

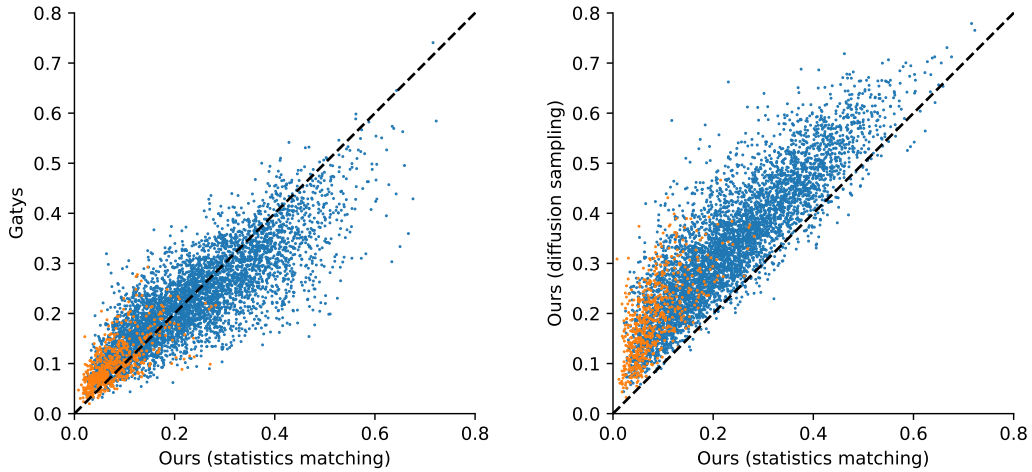


Figure 3: Perceptual quality evaluated using DreamSim (smaller is better). Each point corresponds to one image in the DTD dataset. Color labeling designates homogeneity of each image, compared with the threshold used to select the training set (orange: more homogeneous, blue: less homogeneous). Left: our model with statistics matching v.s. the Gatys model; right: our model with statistics matching v.s. with diffusion sampling. In both panels, points above the dashed identity line correspond to images on which our model with statistics matching performs better.

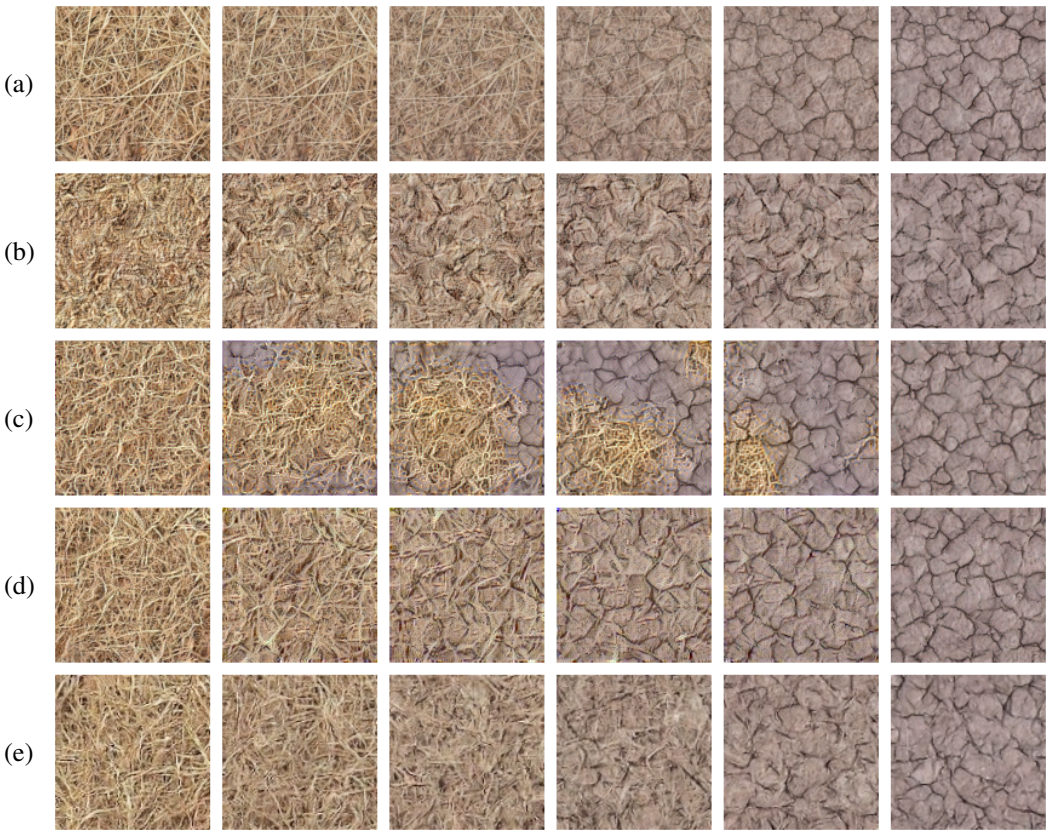


Figure 4: Interpolation between the 5th and 6th images in Figure 2. **(a)** Fading in pixel values. **(b)** Portilla-Simoncelli model. **(c)** Gatys model. **(d)** Statistics matching for our model. **(e)** Diffusion sampling for our model.

textures, then these examples provide evidence that the representation space of the Gatys model is not convex. In contrast, interpolated samples from the Portilla-Simoncelli model and from our model (interpolated in μ or λ) are seen to be homogeneous, with features that smoothly transition from those of the two endpoint textures.[4] For μ-space interpolation via statistics matching, samples sometimes contain sporadic artifacts, especially at their boundaries (this is present in Figure 4 but more visible in Figure 13). For interpolation of λ-space via diffusion, such artifacts are not present, although the quality is often a bit worse than that of statistics matching (as is shown in Figure 2).

4 Discussion

We have developed a framework for learning a maximum entropy density model from data. Specifically, given a training set of texture images, we learn an exponential family model that is defined in terms of a set of statistics, such that each image corresponds to a sample from the maximum entropy density constrained to match those statistics. We use a two-network architecture (one computing the statistics, and other the weights) in which their combination is an inner product, as dictated by the maximum entropy construction: the log probability is a weighted sum of the statistics. The full model is optimized for denoising performance, as is done for generative diffusion models.

We find that images generated by our model trained on a large dataset of textures are visually superior to the state-of-the-art Gatys model, while relying on a far smaller parameter set (512 vs. 176,640). Our model formulation enables two generative sampling methods: statistics matching and diffusion sampling, and we find that the former generates higher quality samples. We suspect that this may be related to the hypothesis in our mathematical framework that the exponential family distribution needs to be closed under additive Gaussian noise, which is generally not exactly true. Deviation from this hypothesis may hurt diffusion sampling more than statistics matching, since diffusion sampling depends on the accurate estimation of scores at every noise level, while statistics matching only aims to match the statistics for the *clean* image and is adaptive during the optimization process (see Appendix C).

We also show that a straight trajectory between the representations (μ or λ) of two given textures generates a visually smooth sequence of interpolated textures, suggesting that the representation space of our model has convexity properties that are desirable for texture manipulation (e.g., style transfer). We are also interested in discovering interpretable or semantically meaningful directions in the representation space (as in Hudson et al. [2024]), either with unsupervised dimensionality reduction methods, or using category labels in datasets such as DTD [Cimpoi et al., 2014].

Our texture model can serve as a scientific tool for generating stimuli for neuroscience and psychology experiments [Victor et al., 2017]. Several existing papers [Freeman et al., 2013, Lieber et al., 2025] are based on the texture model in Portilla and Simoncelli [2000], which is relatively low-dimensional (710 statistics for grayscale images, $\sim$3k for color images) and more interpretable by construction, but is worse in visual quality than those based on deep neural networks. Jagadeesh and Gardner [2022] uses the Gatys model to generate stimuli for experiments, which has a very large dimensionality ($\sim$177k statistics). With a much smaller size (512 statistics), our model offers a decisive improvement in both model size and visual quality, opening the possibility of characterizing neural responses in the representation space.

Finally, our method is general. As with generative diffusion models, it relies on the use of networks with inductive biases that are well-matched to the target density, and can be applied to any data modality that can be appropriately described by a maximum entropy model, such as homogeneous temporal segments of sounds [McDermott and Simoncelli, 2011], spatiotemporal patches of videos [Xie et al., 2019], or spiking data measured from biological neurons [Schneidman et al., 2006, Mayzel and Schneidman, 2024, Lynn et al., 2025]. Learning statistics directly from the data could lead to improved models for these modalities, as it has for visual textures.

[4] Note that the original Portilla-Simoncelli paper [Portilla and Simoncelli, 2000] reported that samples for interpolated parameters appeared as patchwise spatial mixtures (like the Gatys model), but those examples were not computed by linearly interpolating the model parameters. Instead, the parameters were transformed to expected values (e.g., correlations are replaced by unnormalized and uncentered second moments) and linear interpolation was performed there. The rationale was to guarantee the existence of images that are consistent with the interpolated parameters. Linear interpolation on these nonlinearly transformed parameters is not equivalent to linear interpolation on the original parameters, as was done for the examples in row (b) of Figures 4, 12 and 13.

Acknowledgments and Disclosure of Funding

We would like to thank Florentin Guth for suggesting the twin half-UNet architecture of the f network (bottom half of Figure 1).

This work was supported by the Flatiron Institute at the Simons Foundation.

References

David J Heeger and James R Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238, 1995.

Jeremy De Bonet and Paul Viola. A non-parametric multi-scale statistical model for natural images. *Advances in neural information processing systems*, 10, 1997.

Alexei A Efros and William T Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 341–346, 2001.

Song Chun Zhu, Yingnian Wu, and David Mumford. Filters, random fields and maximum entropy (FRAME): Towards a unified theory for texture modeling. *International Journal of Computer Vision*, 27(2):107–126, 1998.

Javier Portilla and Eero P Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International journal of computer vision*, 40:49–70, 2000.

Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. *Advances in neural information processing systems*, 28, 2015.

Yang Lu, Song-Chun Zhu, and Ying Wu. Learning FRAME models using CNN filters. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE transactions on pattern analysis and machine intelligence*, 44(5):2567–2581, 2020.

Valentin De Bortoli, Agnes Desolneux, Alain Durmus, Bruno Galerne, and Arthur Leclaire. Maximum entropy methods for texture synthesis: Theory and practice. *SIAM Journal on Mathematics of Data Science*, 3(1):52–82, 2021.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International conference on learning representations*, 2018.

Jeremy Freeman, Corey M Ziemba, David J Heeger, Eero P Simoncelli, and J Anthony Movshon. A functional and perceptual signature of the second visual area in primates. *Nature neuroscience*, 16 (7):974–981, 2013.

Yunguo Yu, Anita M Schmid, and Jonathan D Victor. Visual processing of informative multipoint correlations arises primarily in V2. *Elife*, 4:e06604, 2015.

Bela Julesz. Visual pattern discrimination. *IRE transactions on Information Theory*, 8(2):84–92, 1962.

Jonathan D Victor and Mary M Conte. Local image statistics: Maximum-entropy constructions and perceptual salience. *Journal of the Optical Society of America A*, 29(7):1313–1345, 2012.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32, 2019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

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*, pages 10684–10695, 2022.

Zahra Kadkhodaie and Eero Simoncelli. Stochastic solutions for linear inverse problems using the prior implicit in a denoiser. *NeurIPS*, 34:13242–13254, 2021.

Bahjat Kawar, Michael Elad, Stefano Ermon, and Jiaming Song. Denoising diffusion restoration models. 35:23593–23606, 2022.

Hyungjin Chung, Jeongsol Kim, Michael Thompson Mccann, Marc Louis Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems. In *ICLR*, 2023.

Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G. Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems. *arXiv preprint arXiv:2410.00083*, 2024.

Zahra Kadkhodaie, Stéphane Mallat, and Eero P Simoncelli. Unconditional CNN denoisers contain sparse semantic representation of images. In *Proc 60th Asilomar Conf on Signals, Systems and Computers*, Pacific Grove, CA, Oct 25-28 2026. IEEE Computer Society.

Michael Fuest, Pingchuan Ma, Ming Gui, Johannes Schusterbauer, Vincent Tao Hu, and Björn Ommer. Diffusion models and representation learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026.

Sarthak Mittal, Korbinian Abstreiter, Stefan Bauer, Bernhard Schölkopf, and Arash Mehrjou. Diffusion based representation learning. In *International conference on machine learning*, pages 24963–24982. PMLR, 2023.

Konpat Preechakul, Nattanat Chatthee, Suttisak Wizadwongsa, and Supasorn Suwajanakorn. Diffusion autoencoders: Toward a meaningful and decodable representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10619–10629, 2022.

Yingheng Wang, Yair Schiff, Aaron Gokaslan, Weishen Pan, Fei Wang, Christopher De Sa, and Volodymyr Kuleshov. Infodiffusion: Representation learning using information maximizing diffusion models. In *International Conference on Machine Learning*, pages 36336–36354. PMLR, 2023.

Drew A Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K Lampinen, Andrew Jaegle, James L McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner. SODA: Bottleneck diffusion models for representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23115–23127, 2024.

Ruihan Yang and Stephan Mandt. Lossy image compression with conditional diffusion models. *Advances in Neural Information Processing Systems*, 36:64971–64995, 2023.

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*, pages 248–255. Ieee, 2009.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023.

Herbert E Robbins. An empirical Bayes approach to statistics. In *Proc. Third Berkeley Symp. Math. Statist. Prob.*, pages 388–394. Springer, 1956.

Koichi Miyasawa. An empirical Bayes estimator of the mean of a normal population. *Bull. Inst. Internat. Statist*, 38(181-188), 1961.

Martin Raphan and Eero P Simoncelli. Least squares estimation without priors or supervision. *Neural Comp.*, 23(2):374–420, 2011.

Aapo Hyvärinen and Peter Dayan. Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, 6(4), 2005.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.

Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3606–3613, 2014.

Gertjan J Burghouts and Jan-Mark Geusebroek. Material-specific adaptation of color invariant features. *Pattern Recognition Letters*, 30(3):306–313, 2009.

Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on graphics (TOG)*, 32(4):1–17, 2013.

Eero P Simoncelli and William T Freeman. The steerable pyramid: A flexible architecture for multi-scale derivative computation. In *ICIP (3)*, pages 444–447, 1995.

Lyndon Duong, Kathryn Bonnen, William Broderick, Pierre-Étienne Fiquet, Nikhil Parthasarathy, Thomas Yerxa, Xinyuan Zhao, and Eero Simoncelli. Plenoptic: A platform for synthesizing model-optimized visual stimuli. *Journal of Vision*, 23(9):5822–5822, 2023.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, et al. Pytorch 2: Faster machine learning through dynamic python bytecode transformation and graph compilation. In *Proceedings of the 29th ACM international conference on architectural support for programming languages and operating systems, volume 2*, pages 929–947, 2024.

Michel Talagrand. A new look at independence. *The Annals of probability*, pages 1–34, 1996.

Yifan Wang, Aleksander Holynski, Brian L Curless, and Steven M Seitz. Infinite texture: Text-guided high resolution diffusion texture synthesis. *arXiv preprint arXiv:2405.08210*, 2024.

Julia Guerrero-Viu, Milos Hasan, Arthur Roullier, Midhun Harikumar, Yiwei Hu, Paul Guerrero, Diego Gutierrez, Belen Masia, and Valentin Deschaintre. Texsliders: Diffusion-based texture editing in clip space. In *ACM SIGGRAPH 2024 conference papers*, pages 1–11, 2024.

Alexander Mordvintsev and Eyvind Niklasson. μNCA: Texture generation with ultra-compact neural cellular automata. *arXiv preprint arXiv:2111.13545*, 2021.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pages 586–595. IEEE, 2018.

Jonathan D Victor, Mary M Conte, and Charles F Chubb. Textures as probes of visual processing. *Annual review of vision science*, 3:275–296, 2017.

Justin D Lieber, Timothy D Oleskiw, Laura Palmieri, Eero P Simoncelli, and J Anthony Movshon. Responses of neurons in macaque V4 to object and texture images. *BioRxiv*, pages 2024–02, 2025.

Akshay V Jagadeesh and Justin L Gardner. Texture-like representation of objects in human visual cortex. *Proceedings of the National Academy of Sciences*, 119(17):e2115302119, 2022.

Josh H McDermott and Eero P Simoncelli. Sound texture perception via statistics of the auditory periphery: Evidence from sound synthesis. *Neuron*, 71(5):926–940, 2011.

Jianwen Xie, Song-Chun Zhu, and Ying Nian Wu. Learning energy-based spatial-temporal generative convnets for dynamic patterns. *IEEE transactions on pattern analysis and machine intelligence*, 43(2):516–531, 2019.

Elad Schneidman, Michael J Berry, Ronen Segev, and William Bialek. Weak pairwise correlations imply strongly correlated network states in a neural population. *Nature*, 440(7087):1007–1012, 2006.

Jonathan Mayzel and Elad Schneidman. Homeostatic synaptic normalization optimizes learning in network models of neural population codes. *Elife*, 13:RP96566, 2024.

Christopher W Lynn, Qiwei Yu, Rich Pang, Stephanie E Palmer, and William Bialek. Exact minimax entropy models of large-scale neuronal activity. *Physical Review E*, 111(5):054411, 2025.

Zhou Wang and Eero P Simoncelli. Maximum differentiation (MAD) competition: A methodology for comparing computational models of perceptual quantities. *Journal of vision*, 8(12):8–8, 2008.

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. *Advances in neural information processing systems*, 30, 2017.

Tuomas Kynkäänniemi, Tero Karras, Miika Aittala, Timo Aila, and Jaakko Lehtinen. The role of ImageNet classes in Fréchet inception distance. *arXiv preprint arXiv:2203.06026*, 2022.

Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. *arXiv preprint arXiv:1801.01401*, 2018.

Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in neural information processing systems*, 32, 2019.

Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking FID: Towards a better evaluation metric for image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9307–9315, 2024.

A Maximum entropy probability distribution

A.1 Derivation of the exponential family

We rewrite (1) as:

$$\min_p \int p(x) \log p(x) dx \quad \text{s.t.} \int p(x) dx = 1, \int p(x) f_k(x) dx = \mu_k, \quad k = 1, 2, \cdots, d \tag{13}$$

Combining the objective and constraints using the method of Lagrange multipliers gives a combined objective:

$$\int p(x) \log p(x) dx + \nu \left(\int p(x) dx - 1 \right) + \sum_{k=1}^{d} \lambda_k \left(\int p(x) f_k(x) dx - \mu_k \right) \tag{14}$$

Taking the functional derivative with respect to p and setting it to zero gives:

$$\log p(x) + 1 + \nu + \sum_{k=1}^{d} \lambda_k f_k(x) = 0 \tag{15}$$

$$\Rightarrow p(x) = \exp \left(-1 - \nu - \sum_{k=1}^{d} \lambda_k f_k(x) \right) \tag{16}$$

Since one of the constraints is that p is normalized, we can eliminate ν:

$$\exp(1 + \nu) = \int \exp \left(- \sum_{k=1}^{d} \lambda_k f_k(x) \right) dx =: Z(\lambda) \tag{17}$$

Then we arrive at an exponential family distribution (2):

$$p(x) = \frac{1}{Z(\lambda)} \exp \left(- \sum_{k=1}^{d} \lambda_k f_k(x) \right) \tag{18}$$

The rest of the Lagrange multipliers $\{\lambda_1, \lambda_2, \cdots, \lambda_d\}$ are chosen to satisfy the other constraints, and are thus a function of the statistical measurements $\{\mu_1, \mu_2, \cdots \mu_d\}$. In most cases, these relationships cannot be expressed in closed form (the Gaussian distribution is an exception, as we show below).

A.2 Example: the Gaussian distribution

Here, we consider the simplest example of maximum-entropy density model. Suppose x is a scalar random variable, and one chooses to constrain the density using two statistics $f(x) = [x, x^2]$ (i.e., the first and second moments). Then the maximum entropy distribution is:

$$p_\lambda(x) = \frac{1}{Z(\lambda)} \exp(-\lambda_1 x - \lambda_2 x^2) \tag{19}$$

The parameters λ_1 and λ_2 must be chosen to satisfy the constraints $\mathbb{E}(x) = \mu_1$ and $\mathbb{E}(x^2) = \mu_2$. In general, $\lambda = [\lambda_1, \lambda_2]$ cannot be written analytically as a function of $\mu = [\mu_1, \mu_2]$, but in this case we see that the density is an exponential of a quadratic form, and thus a Gaussian distribution. So we can solve directly, giving:

$$\lambda_1 = -\frac{\mu_1}{\mu_2 - \mu_1^2}, \qquad \lambda_2 = \frac{1}{2(\mu_2 - \mu_1^2)} \tag{20}$$

Note that $\mu_2 - \mu_1^2$ is the variance (second moment minus squared first moment). The mapping between μ and λ is one-to-one, and either can serve as a parametrization of the family of Gaussian distributions.

A.3 One-to-one correspondence between λ and μ

Define the set of λ that leads to a normalizable distribution:

$$\Lambda = \{\lambda \in \mathbb{R}^d : Z(\lambda) < +\infty\} \tag{21}$$

We also know that the set of μ that is achievable by some p is:

$$M = \{\mu \in \mathbb{R}^d : \mu = \mathbb{E}_{x\sim p}[f(x)], \exists p\} = \text{conv}(\{f(x) : x \in \mathbb{R}^n\}) \tag{22}$$

where conv denotes the convex hull.

It is always true that λ uniquely determines μ. When the following two conditions are met:

- minimal: there is no λ such that $\sum_{i=1}^d \lambda_k f_k(x) =$ constant.
- regular: Λ is an open set (true for all useful distributions).

Then it is also true that any μ in the interior of M uniquely determines λ. In other words, we have a one-to-one correspondence (bijection) between Λ and interior(M). [5]

B Tweedie's formula for a variance-preserving noise model

Consider the variance-preserving noise process:

$$y = \frac{1}{\sqrt{\sigma^2+1}}x + \frac{\sigma}{\sqrt{\sigma^2+1}}\varepsilon \tag{23}$$

where ε is Gaussian noise with idenity covariance. Then

$$p(y|x) = \mathcal{N}\left(y; \frac{1}{\sqrt{\sigma^2+1}}x, \frac{\sigma^2}{\sigma^2+1}I\right) \propto \exp\left(-\frac{\sigma^2+1}{2\sigma^2}\left\|y - \frac{1}{\sqrt{\sigma^2+1}}x\right\|^2\right) \tag{24}$$

Then we compute the score (i.e. the gradient of log probability):

$$\begin{aligned}
\nabla_y \log p(y) &= \frac{1}{p(y)}\nabla_y p(y) && (25)\\
&= \frac{1}{p(y)}\int p(x)\nabla_y p(y|x)dx && (26)\\
&= \frac{1}{p(y)}\int p(x)p(y|x)\left(-\frac{\sigma^2+1}{\sigma^2}\right)\left(y - \frac{1}{\sqrt{\sigma^2+1}}x\right)dx && (27)\\
&= -\int \frac{p(x)p(y|x)}{p(y)}\left(\frac{\sqrt{\sigma^2+1}}{\sigma}\right)\left(\frac{\sqrt{\sigma^2+1}}{\sigma}y - \frac{1}{\sigma}x\right)dx && (28)\\
&= -\frac{\sqrt{\sigma^2+1}}{\sigma}\int p(x|y)\varepsilon(x,y)dx && (29)\\
&= -\frac{\sqrt{\sigma^2+1}}{\sigma}\mathbb{E}[\varepsilon|y] && (30)
\end{aligned}$$

C Relationship between statistics matching and diffusion sampling

In statistics matching of $f : \mathbb{R}^n \to \mathbb{R}^d$ for a given base image $x_0 \in \mathbb{R}^n$, we optimize:

$$\arg\min_x \|f(x) - f(x_0)\|^2 \tag{31}$$

If we use standard gradient descent (instead of Adam, which was used to generate samples in all experiments), the update step is:

$$x \leftarrow x - \beta J_f(x)^T[f(x) - f(x_0)] \tag{32}$$

[5] See `https://www.stat.umn.edu/geyer/5421/notes/expfam.pdf` for a mathematically rigorous treatment.

where $\beta > 0$ is the learning rate, and $J_f : \mathbb{R}^n \to \mathbb{R}^{d \times n}$ is the Jacobian matrix of f.

In the DDPM procedure of (11), an update step in diffusion sampling moves the image x in a direction that is a linear combination of x and the ε-predictor $\hat{\varepsilon}(x; \lambda)$, where $\lambda = \lambda(x_0, \sigma_t)$ at each time step t is determined by the base image x_0 and noise schedule σ_t. According to (8), the ε-predictor is a linear combination of x and the output of the network, which is:

$$\nabla_x [\lambda^T f(x)] = J_f(x)^T \lambda(x_0, \sigma_t) \tag{33}$$

Together, these mean that the update step for diffusion sampling moves x in the subspace spanned by x and the rows of $J_f(x)$. If we assume $x \in \text{span}\{\text{rows of } J_f(x)\}$ (the same assumption we made to justify the reparameterization of v-prediction), then we can rearrange terms in the update rule, such that there is a $\tilde{\lambda}$ (determined by x_0 and the noise schedule σ_t for each time step t) such that the update rule reads:

$$x \leftarrow x - J_f(x)^T \tilde{\lambda}(x_0, \sigma_t) + \text{noise} \tag{34}$$

Both update rules move x in a direction that lies in the subspace spanned by the rows of $J_f(x)$. Apart from the additive noise, the fundamental difference between the two rules is that the vector-Jacobian product with $J_f(x)$ (which linearly combines its rows) is done with $f(x) - f(x_0)$ in statistics matching, and with $\tilde{\lambda}(x_0, \sigma_t)$ in diffusion sampling. The former is adaptive to the value of x during the optimization process, while the latter is predetermined for each step.

D Additional results

D.1 Competitive adversarial comparison

In order to more directly compare two generative texture models based on statistical measurements f and g, we let them directly compete to expose each other's flaws, using a method similar to MAD competition [Wang and Simoncelli, 2008]. Given a texture image x_0, we synthesize an image x that has the same representation as x_0 according to one model, but is maximally different according to the other:

$$\max_x \|f(x) - f(x_0)\|^2 \qquad \text{s.t. } g(x) = g(x_0). \tag{35}$$

In practice, we optimize:

$$\max_x \left[\log \left(\frac{\|f(x) - f(x_0)\|^2}{\|f(x_0)\|^2} + \delta \right) - \log \left(\frac{\|g(x) - g(x_0)\|^2}{\|g(x_0)\|^2} + \delta \right) \right], \tag{36}$$

and we also optimize the same objective but with f and g switched. In order to stabilize the training and make sure the constraint is met, we find it helpful to run optimization with only the second term (equivalent to statistics matching) for 500 steps before using the full loss function for another 4500 steps. The constraint is adequately enforced when using a small $\delta = 10^{-5}$. To determine which model is superior, we compared the two generated images in terms of their perceptual quality and deviation from the original texture x_0.

We apply this method to compare the Gatys model and our own (Figure 5; additional examples in Figure 11). Row (b) is dominated by jarring high frequency artifacts, showing that the standard Gatys model does not constrain the high frequency component of the image. To prevent the objective from exclusively exploiting this type of distortion, we add a single high-pass statistic (taken from the steerable pyramid [Simoncelli and Freeman, 1995]) to the Gatys model. When the Gatys model is complemented with this high-pass statistic, it prevents the high frequency artifacts instead allows the appearance of localized gratings in some textures, and changes in local luminance for mosaic-like textures (row (c)). Our model shows its blind spot in row (d), where it allows for a particular type of artifact made of localized oriented structures for most test textures.

D.2 FID score

We quantitatively evaluated the quality and diversity of the generated textures using the FID score [Heusel et al., 2017]. To do this, we use several high-resolution texture images [6] with a large number

[6] Downloaded from `unsplash.com`

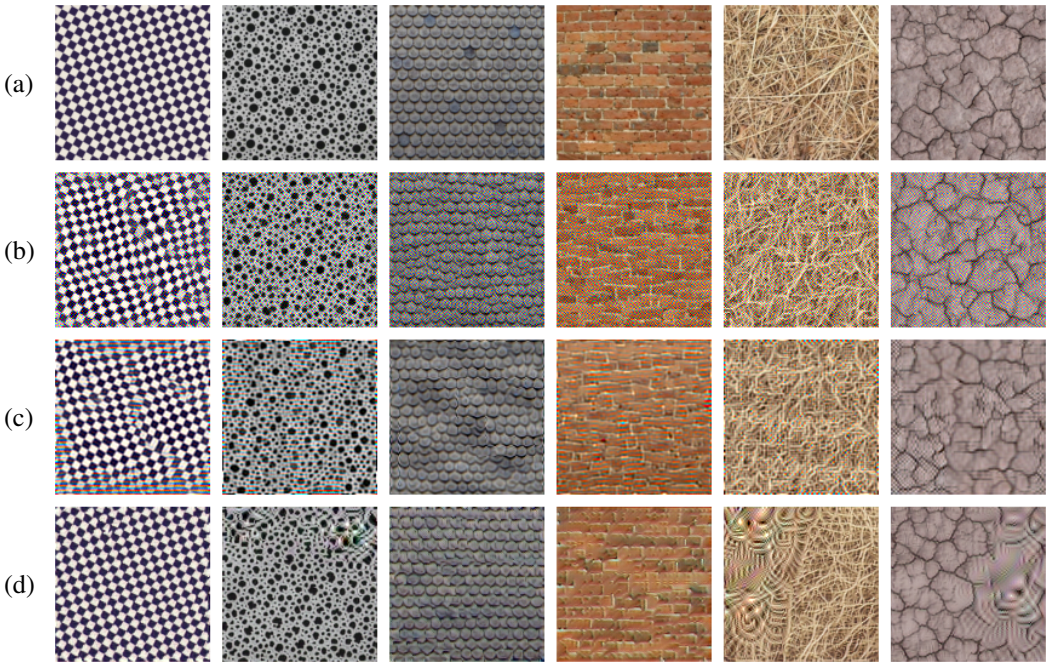


Figure 5: Competitive adversarial comparison between the Gatys model and our model. **(a)** Original texture images (as in Figure 2). **(b)** Images that are considered the same as (a) according to the Gatys model and maximally different according to our model (zoom in to see the high frequency details). **(c)** Same as (b) but adding a high-pass statistic to the Gatys model. **(d)** Images that are considered the same as (a) according to our model and maximally different according to the Gatys model.

Table 2: FID scores. From top to bottom: statistics matching for the Gatys model, statistics matching for our model, diffusion sampling for our model.

	Grass	Pebble	Star	Cloth	Rug	Flower	Marble	Rubber	Glitter
Gatys	175.03	**169.35**	209.10	126.79	273.30	74.03	135.17	108.01	96.19
Stat.	**102.58**	214.77	68.19	**41.70**	105.96	80.49	137.79	**102.46**	**20.82**
Diff.	221.30	192.94	**48.78**	66.25	**55.49**	**67.14**	**46.68**	142.12	36.82

of repeating units, so that we can generate a large number of crops, all of which are homogeneous textures of the same class. For each image, we compute the representation on 10,000 randomly cropped patches, average over them to obtain one single representation, and resample 10,000 patches based on this representation using the aforementioned methods. The FID score is then calculated against the original 10,000 crops (Table 2). For these 8 of the 9 images, the best-performing model is either statistics matching or diffusion sampling. We note that the scores for both our model and the Gatys model are significantly worse than the state-of-the-art values for common data sets (such as ImageNet), but FID scores are not directly comparable across data sets. Moreover, the FID score is constructed using a network trained for object recognition, and is closely tied to the object categories of ImageNet. It has been shown to provide a poor measure of visual quality when used on a model that generates images containing objects outside the categories of ImageNet [Kynkäänniemi et al., 2022]. There are many alternative metrics [Bińkowski et al., 2018, Kynkäänniemi et al., 2019, Jayasumana et al., 2024], but they are also based on networks trained on a specific task (in most cases, object recognition), and on corresponding images that are out-of-distribution for our texture model. In general, we lack a general method for evaluating both quality and diversity of generative models for any data distribution.

Table 3: DreamSim, LPIPS and DISTS perceptual metrics. Mean and standard deviation are computed over 10,000 samples in each table entry. Smaller number is better (more similar).

		Two crops	Gatys	Stat. matching	Diffusion
DreamSim	Grass	**0.198 ± 0.060**	0.294 ± 0.042	0.273 ± 0.043	0.268 ± 0.058
	Pebble	**0.176 ± 0.046**	0.224 ± 0.037	0.332 ± 0.049	0.354 ± 0.063
	Star	**0.101 ± 0.042**	0.208 ± 0.029	0.143 ± 0.028	0.122 ± 0.034
	Cloth	**0.078 ± 0.034**	0.131 ± 0.033	0.091 ± 0.031	0.113 ± 0.032
	Rug	**0.119 ± 0.037**	0.232 ± 0.026	0.121 ± 0.024	0.143 ± 0.031
	Flower	0.103 ± 0.034	0.118 ± 0.030	**0.094 ± 0.029**	0.134 ± 0.033
	Marble	**0.143 ± 0.034**	0.211 ± 0.028	0.172 ± 0.028	0.193 ± 0.035
	Rubber	**0.134 ± 0.031**	0.189 ± 0.026	0.173 ± 0.028	0.200 ± 0.030
	Glitter	0.118 ± 0.042	0.136 ± 0.031	**0.115 ± 0.033**	0.136 ± 0.039
LPIPS	Grass	0.288 ± 0.022	**0.272 ± 0.019**	0.281 ± 0.019	0.284 ± 0.020
	Pebble	0.436 ± 0.033	**0.429 ± 0.029**	0.441 ± 0.028	0.456 ± 0.029
	Star	0.345 ± 0.046	0.337 ± 0.033	**0.336 ± 0.036**	**0.336 ± 0.037**
	Cloth	0.130 ± 0.022	0.132 ± 0.017	**0.128 ± 0.017**	0.135 ± 0.021
	Rug	0.244 ± 0.036	0.245 ± 0.023	**0.236 ± 0.021**	0.266 ± 0.023
	Flower	0.307 ± 0.027	**0.299 ± 0.022**	0.302 ± 0.021	0.307 ± 0.022
	Marble	0.342 ± 0.028	**0.325 ± 0.024**	0.343 ± 0.025	0.341 ± 0.027
	Rubber	0.299 ± 0.023	**0.293 ± 0.019**	**0.293 ± 0.020**	0.298 ± 0.021
	Glitter	0.312 ± 0.030	**0.301 ± 0.024**	0.304 ± 0.023	0.306 ± 0.023
DISTS	Grass	0.273 ± 0.021	**0.271 ± 0.016**	0.290 ± 0.018	0.314 ± 0.024
	Pebble	**0.245 ± 0.022**	0.268 ± 0.017	0.277 ± 0.018	0.293 ± 0.022
	Star	**0.239 ± 0.034**	0.278 ± 0.023	0.268 ± 0.022	0.251 ± 0.021
	Cloth	0.207 ± 0.022	0.207 ± 0.016	**0.205 ± 0.017**	0.220 ± 0.019
	Rug	0.254 ± 0.030	0.261 ± 0.020	**0.247 ± 0.021**	0.261 ± 0.020
	Flower	0.219 ± 0.023	0.219 ± 0.015	**0.217 ± 0.016**	**0.217 ± 0.015**
	Marble	**0.212 ± 0.015**	0.216 ± 0.014	0.226 ± 0.015	0.217 ± 0.014
	Rubber	**0.246 ± 0.018**	0.257 ± 0.014	0.265 ± 0.015	0.295 ± 0.017
	Glitter	0.233 ± 0.019	0.243 ± 0.018	**0.229 ± 0.016**	0.235 ± 0.015

D.3 Comparison of perceptual metrics

Unlike FID, perceptual metrics such as DreamSim [Fu et al., 2023], LPIPS [Zhang et al., 2018] and DISTS [Ding et al., 2020] quantify the distance between two individual images (instead of two distributions), and computing them do not require a large number of crops of the same texture. However, the images generated for computing FID give us a way to compare the perceptual metrics themselves, by testing whether these metrics recover the human perception that the distance between two crops from the original texture is smaller than the distance between one crop and one generated resampled image. To do this, we randomly pair 10,000 crops from the original images with 10,000 generated resampled images and obtain 10,000 scores per metric per generative method (Gatys, ours with statistics matching, ours with diffusion sampling), and we similarly compute the scores between two sets of independent 10,000 crops (Table 3). We find that DreamSim most consistently rated the distance between two original crops as the smallest. Therefore, we choose to report DreamSim scores in the main text (Figure 3 and Table 1).

E Supplementary figures

Figure 6 provides example images from our training dataset, sorted according to their homogeneity score. Figures 7 and 8 show multiple samples per texture image per method. Figure 9 shows resampling for more texture images (in addition to Figure 2). Figure 10 shows resampling for inhomogeneous images. Figure 11 shows competitive adversarial comparison for more texture images (in addition to Figure 5). Figures 12 and 13 provide additional examples of interpolation (in addition to Figure 4).



Figure 6: Example images from our training dataset, ordered from most to least homogeneous according to the score, in equal steps of ranking.

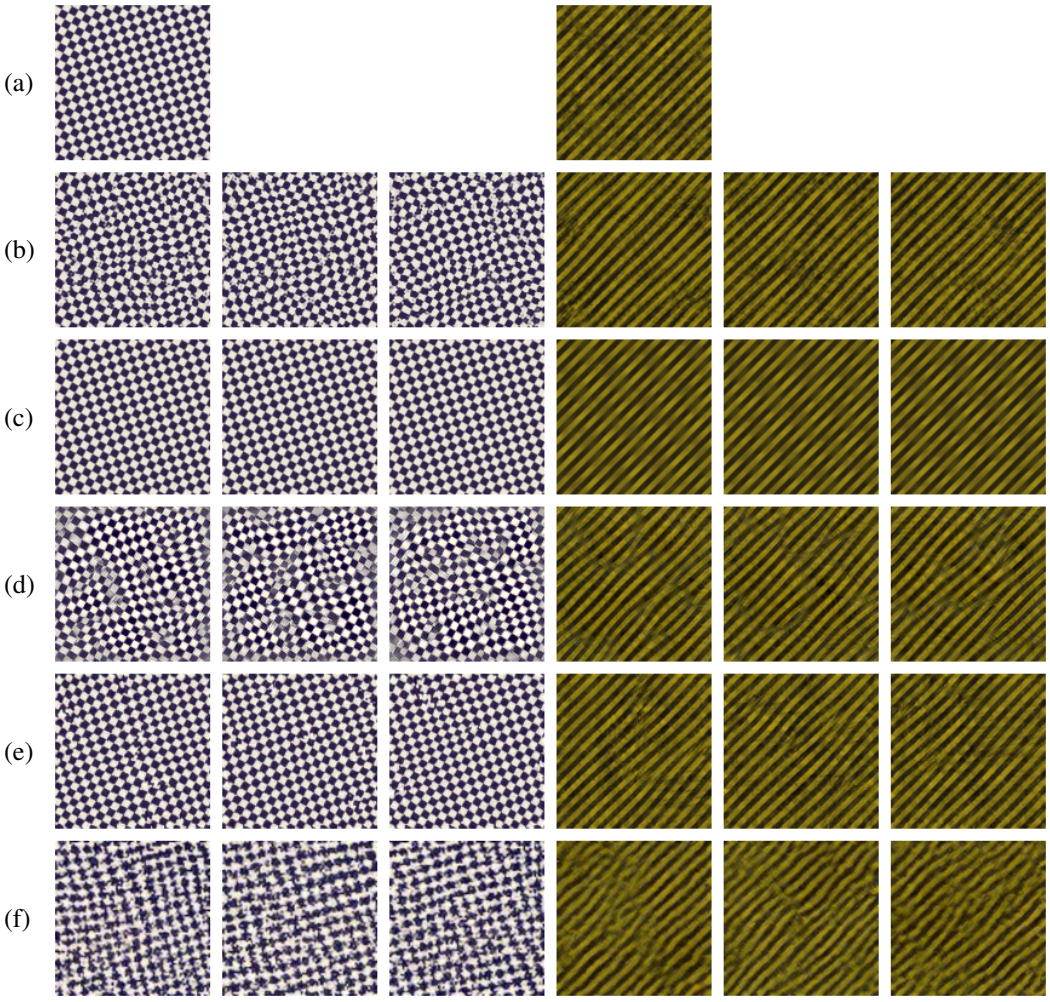


Figure 7: Resampling with three samples per texture image per method. **(a)** Original texture images. **(b)** Portilla-Simoncelli model; **(c)** μNCA model; **(d)** Gatys model; **(e)** statistics matching for our model; **(f)** diffusion sampling for our model.

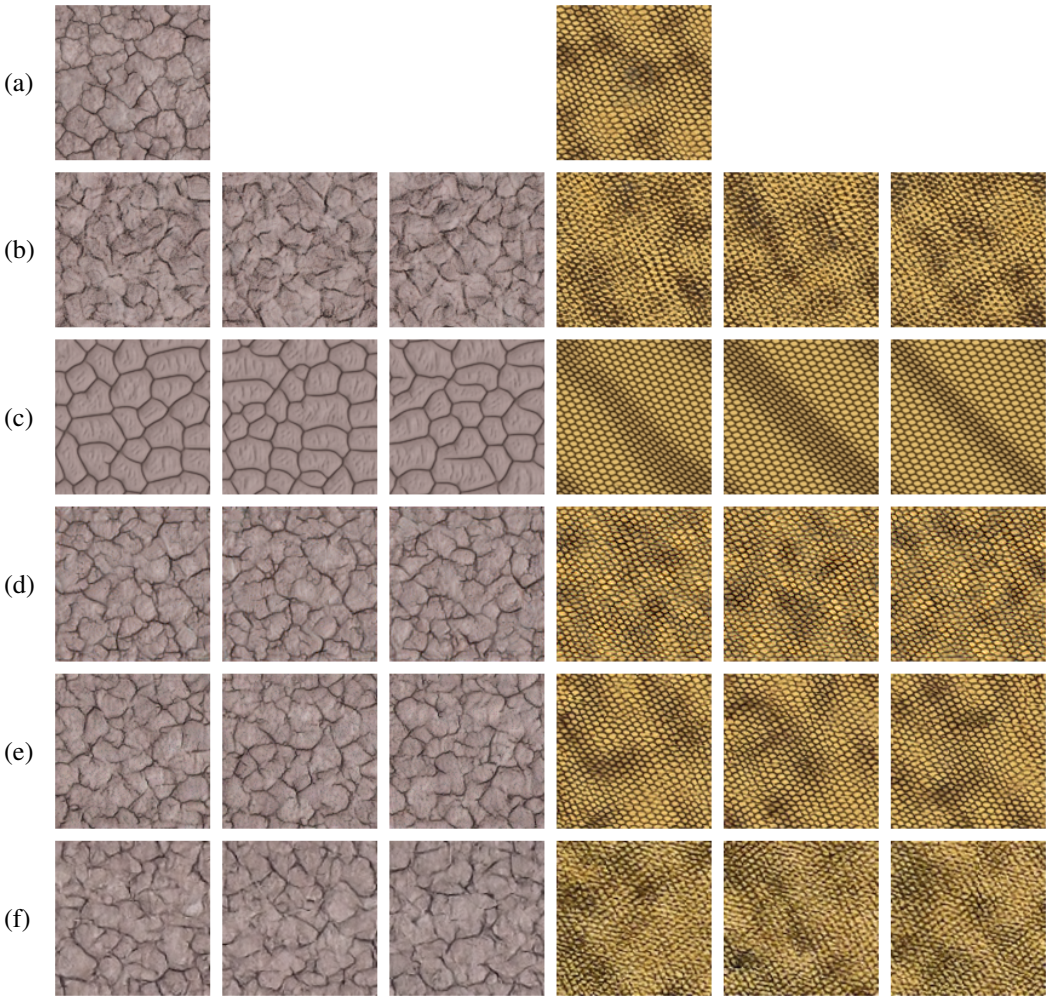


Figure 8: Resampling with three samples per texture image per method. **(a)** Original texture images. **(b)** Portilla-Simoncelli model; **(c)** μNCA model; **(d)** Gatys model; **(e)** statistics matching for our model; **(f)** diffusion sampling for our model.

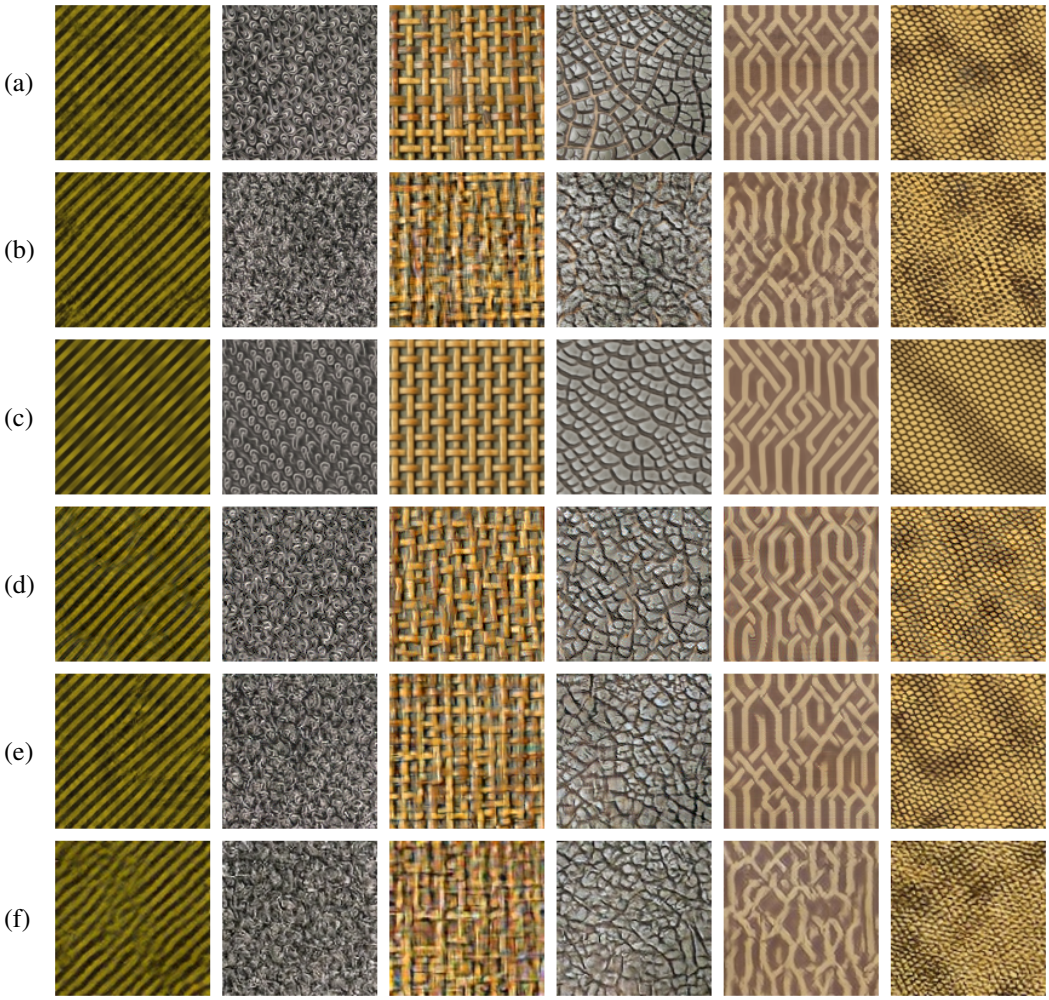


Figure 9: Resampling examples. **(a)** Original texture images (from DTD dataset, sorted in decreasing homogeneity). Remaining rows show samples generated by **(b)** Portilla-Simoncelli model; **(c)** μNCA model; **(d)** Gatys model; **(e)** statistics matching for our model; **(f)** diffusion sampling for our model.



Figure 10: Resampling examples for inhomogeneous images. **(a)** Original texture images (from DTD dataset). Remaining rows show samples generated by **(b)** Portilla-Simoncelli model (a synthesis failure occurred in the fourth panel); **(c)** μNCA model; **(d)** Gatys model; **(e)** statistics matching for our model; **(f)** diffusion sampling for our model.

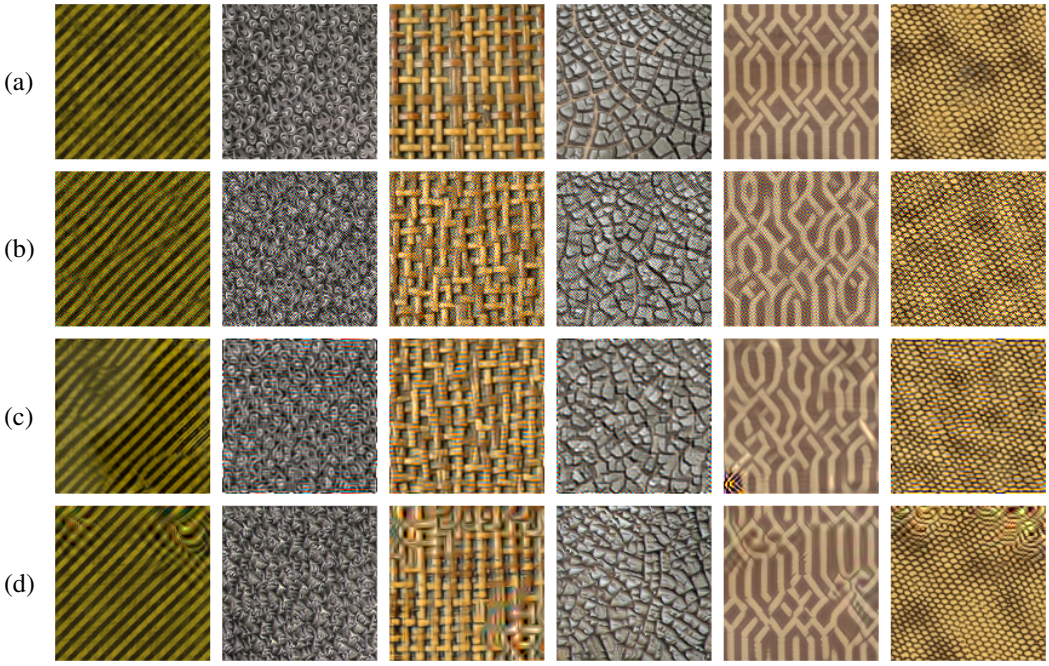


Figure 11: Competitive adversarial comparison between the Gatys model and our model. **(a)** Original texture images (same as Figure 9). **(b)** Images that are considered the same as (a) according to the Gatys model and maximally different according to our model (zoom in to see the high frequency details). **(c)** Same as (b) but adding a high-pass statistic to the Gatys model. **(d)** Images that are considered the same as (a) according to our model and maximally different according to the Gatys model.

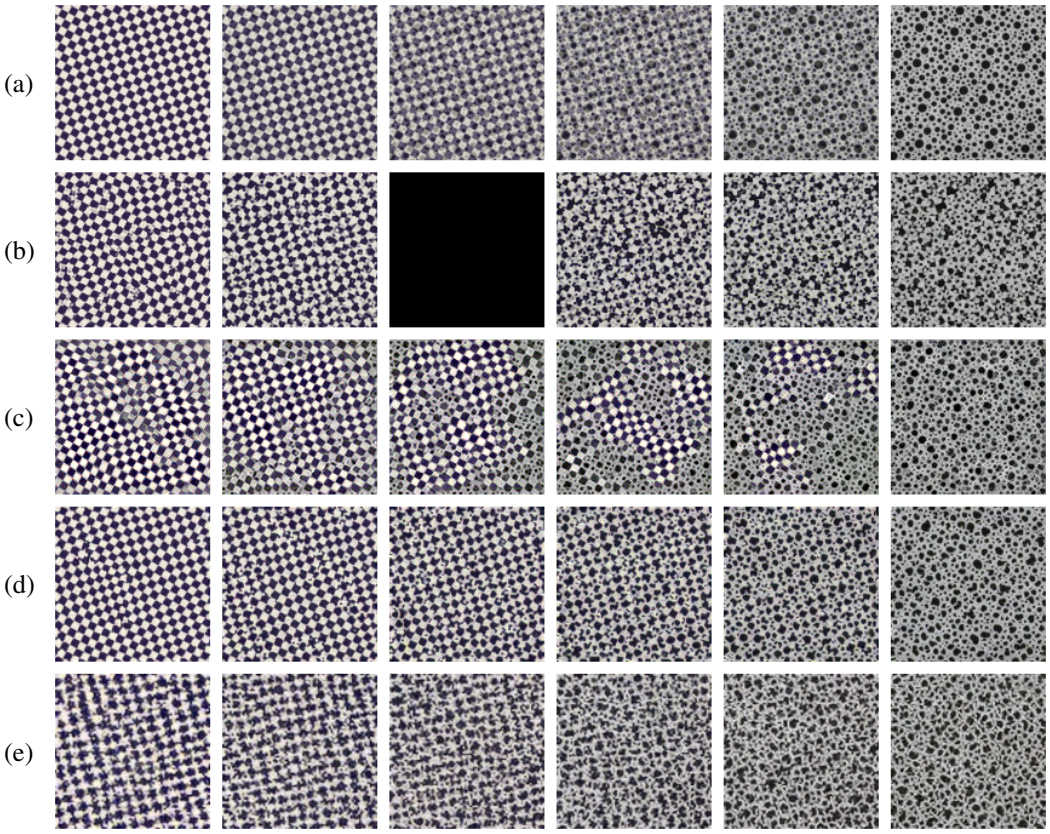


Figure 12: Interpolation between 1st and 2nd images in Figure 2. **(a)** Fading in pixel values. **(b)** Portilla-Simoncelli model (a synthesis failure occurred in the third panel). **(c)** Gatys model. **(d)** Statistics matching for our model. **(e)** Diffusion sampling for our model.

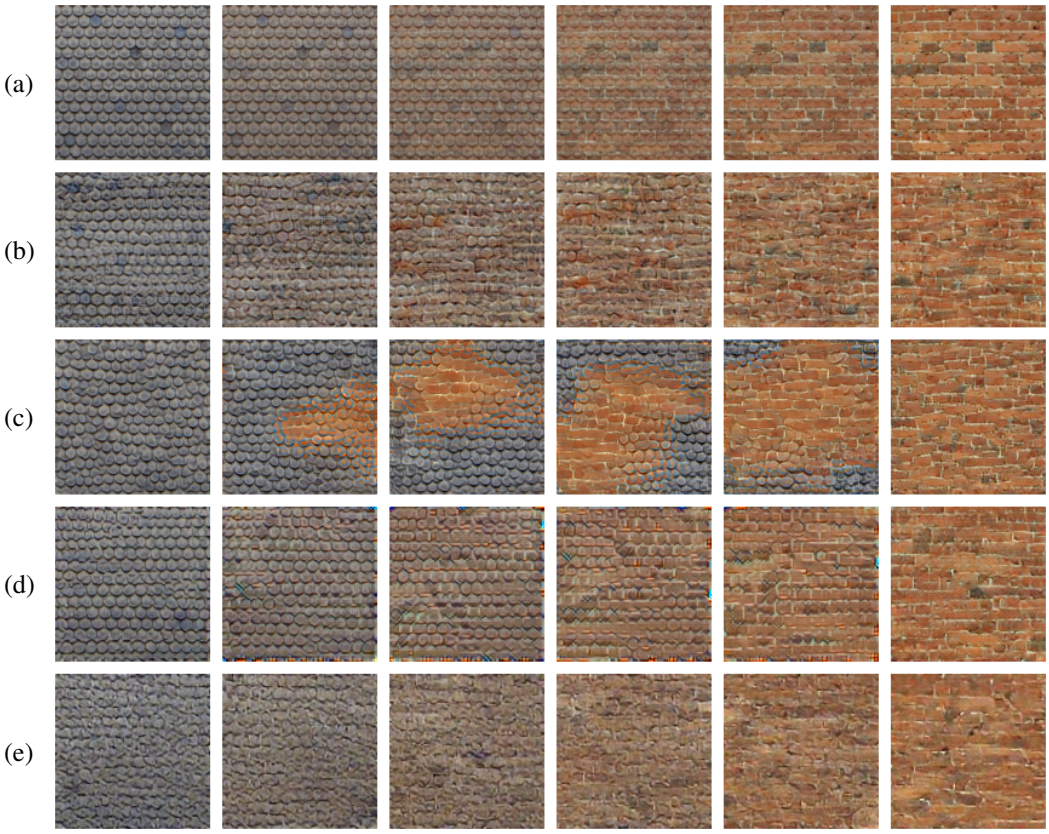


Figure 13: Interpolation between 3rd and 4th images in Figure 2. **(a)** Fading in pixel values. **(b)** Portilla-Simoncelli model. **(c)** Gatys model. **(d)** Statistics matching for our model. **(e)** Diffusion sampling for our model.

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: We described our motivation and contributions clearly in the abstract and introduction.

   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: We discussed the limitation of our mathematical framework and our experiments results in the respective sections and in the discussion.

   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: Derivations for mathematical formulas are found in the appendix.

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: Details of the model architecture, dataset, training, evaluation, etc. are provided in the Method section.

   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: We have released the code that defines and trains the model and produces all results (tables and figures) of this paper, including the code that generates our training dataset.

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: Details of the model architecture, dataset, training, evaluation, etc. are provided in the Method section.

   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: We report mean and standard deviation for perceptual scores and perform appropriate statistical testing for the purpose of comparing two models.

   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: Stated in the Method section.

   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: We followed the NeurIPS Code of Ethics.

   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: [N/A]

    Justification: This paper reports foundational research in computer vision that has no foreseeable societal impact.

    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: The model and datasets in this paper pose no such risks.

    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: ImageNet and DTD are cited.

    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: [Yes]

    Justification: We have released the code that defines and trains the model and produces all results (tables and figures) of this paper, including the code that generates our training dataset.

    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:

    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:

    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: [N/A]

    Justification:

    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.