SketchDNN: Joint Continuous-Discrete Diffusion for CAD Sketch Generation

Sathvik Chereddy, John Femiani

MIAMI UNIVERSITY

Paper Code

Introduction

2D sketch/blueprint design is a tedious and manual aspect of CAD modelling that is an ideal domain for generative AI.

Prior solutions have relied on tokenization and autoregressive approaches, which can't accommodate both the heterogeneous parameterizations nor permutation invariance of primitives.

We propose a novel discrete diffusion method that addresses these limitations through superposition and permutation invariant denoising. Our contributions are namely:

1. The first data-space diffusion model for CAD sketch generation
2. A novel discrete diffusion framework based on the Gaussian-Softmax distribution
3. State-of-the-art results in terms of NLL, FID, and Recall

Gaussian-Softmax Distribution

We introduce the Gaussian-Softmax distribution ($\mathcal{GS}$) as a continuous relaxation of the Categorical distribution, where if $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$ then $\boldsymbol{x} = \text{softmax}\{\boldsymbol{y}\} \sim \mathcal{GS}(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$ with pdf:

$$p(\boldsymbol{y}|\boldsymbol{\mu}, \sigma^2 \boldsymbol{I}) = Z(\sigma)^{-1}\left(\prod_{i=1}^{D} \boldsymbol{y}_i\right) \exp\left(-\frac{1}{2\sigma^2}\left[|\tilde{\boldsymbol{y}} - \boldsymbol{\mu}'|^2 - \frac{1}{D}\left(\mathbf{1}^T(\tilde{\boldsymbol{y}} - \boldsymbol{\mu}')\right)^2\right]\right)$$

where $Z(\sigma) = \sqrt{D(2\pi\sigma^2)^{(D-1)}}$, $\boldsymbol{\mu}' = \boldsymbol{\mu} - (\mu_D)\mathbf{1}$, $\tilde{\boldsymbol{y}} = \log \boldsymbol{y} - (\log y_D)\mathbf{1}$

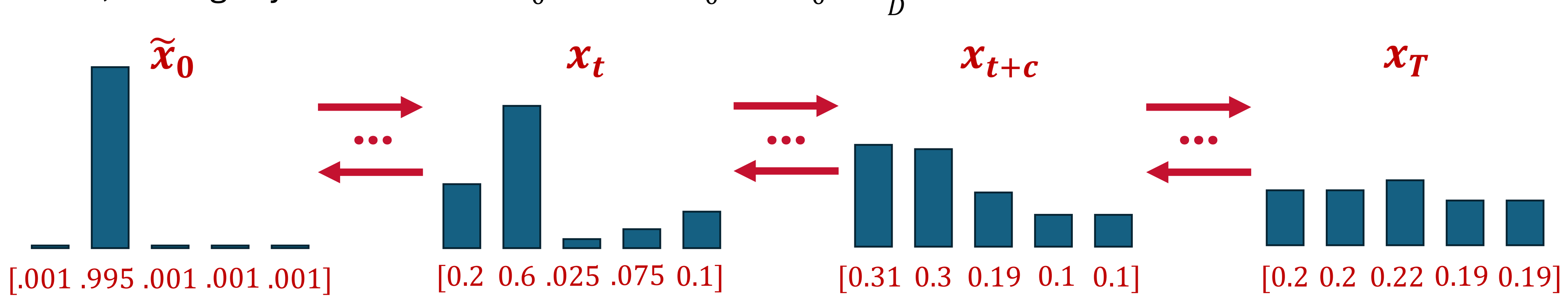
The support of the $\mathcal{GS}$ distribution is the entire probability simplex, unlike the Categorical distribution whose support is only its vertices, which enables $\boldsymbol{x}$ to encode uncertainty.

Discrete Diffusion

Forward: $\boldsymbol{x}_t = \text{softmax}\left\{\sqrt{\bar{b}_t} \log \tilde{\boldsymbol{x}}_0 + \sqrt{\left(1 - \bar{b}_t\right)}\boldsymbol{\epsilon}\right\} \sim \mathcal{GS}\left(\sqrt{\bar{b}_t} \log \tilde{\boldsymbol{x}}_0, \left(1 - \bar{b}_t\right)\boldsymbol{I}\right)$

Reverse: $\boldsymbol{x}_{t-1} = \text{softmax}\left\{\frac{\sqrt{b_t}(1-\bar{b}_{t-1})\log(\tilde{\boldsymbol{x}}_0) + \sqrt{\bar{b}_{t-1}}(1-b_t)\log(\boldsymbol{x}_0^{\theta}(\boldsymbol{x}_t, t))}{1-\bar{b}_t} + \sqrt{\frac{(1-b_t)(1-\bar{b}_{t-1})}{1-\bar{b}_t}}\boldsymbol{\epsilon}\right\}$

Thus, when entropy is maximized at the end of the forward process, the class label follows the uniform distribution i.e., $\text{argmax}\{\boldsymbol{x}_T\} \sim Cat\left(\frac{\mathbf{1}}{D}\right)$. To avoid singularities near $t = 0$, we slightly label smooth $\boldsymbol{x}_0$ so that: $\tilde{\boldsymbol{x}}_0 = k\boldsymbol{x}_0 + \frac{1-k}{D}\mathbf{1}$ where we set $k = .99$



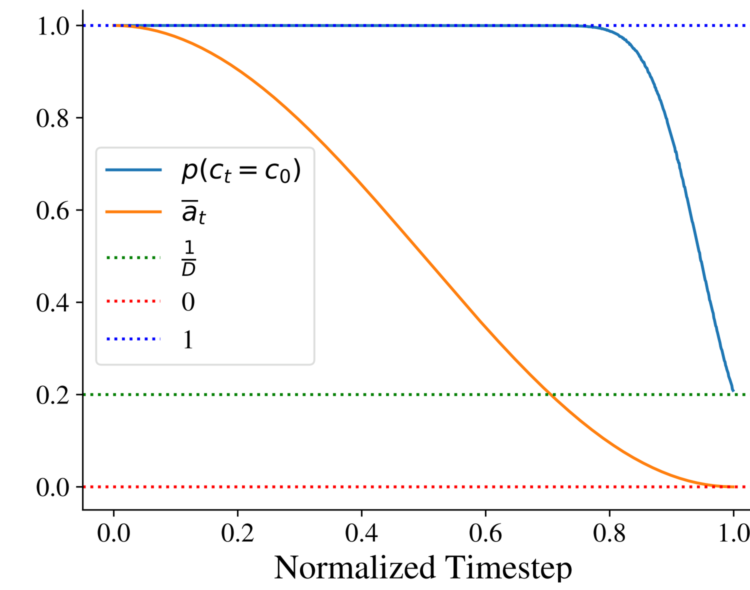
Variance Schedule Augmentation

In Gaussian-Softmax diffusion, we observed that variance schedules cannot be used directly as-is, due to the distortion introduced by the softmax operation on the injected noise. To rectify this, we propose the following variance schedule augmentation:

$$\bar{b}_t = \frac{f(\bar{a}_t)^2}{f(\bar{a}_t)^2 + f(k)^2} \text{ where } f(y) = \log\left(\frac{1-y}{(D-1)y+1}\right)$$

which ensures that the class label is noised according to the chosen schedule $\bar{a}_t$ i.e., $\text{argmax}\{\boldsymbol{x}_t\} \sim Cat\left(\bar{a}_t\boldsymbol{x}_0 + (1 - \bar{a}_t)\frac{\mathbf{1}}{D}\right)$

Raw Schedule



Augmented Schedule

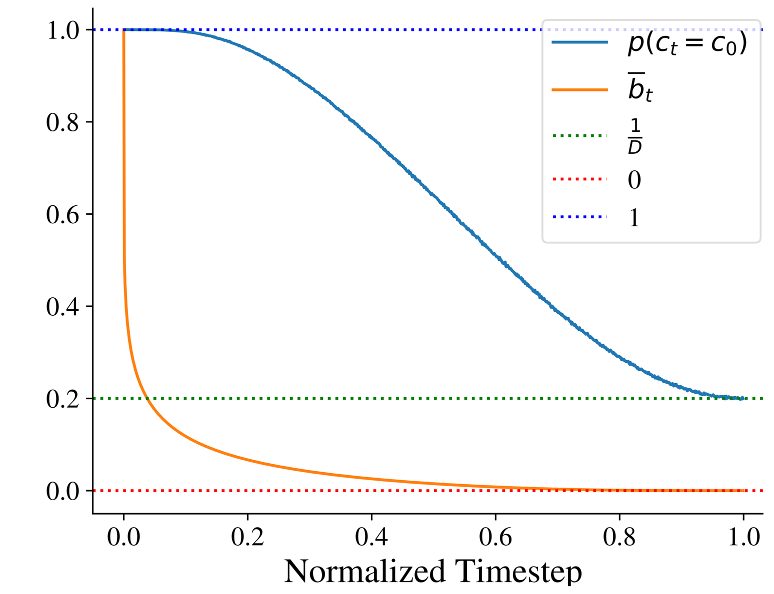
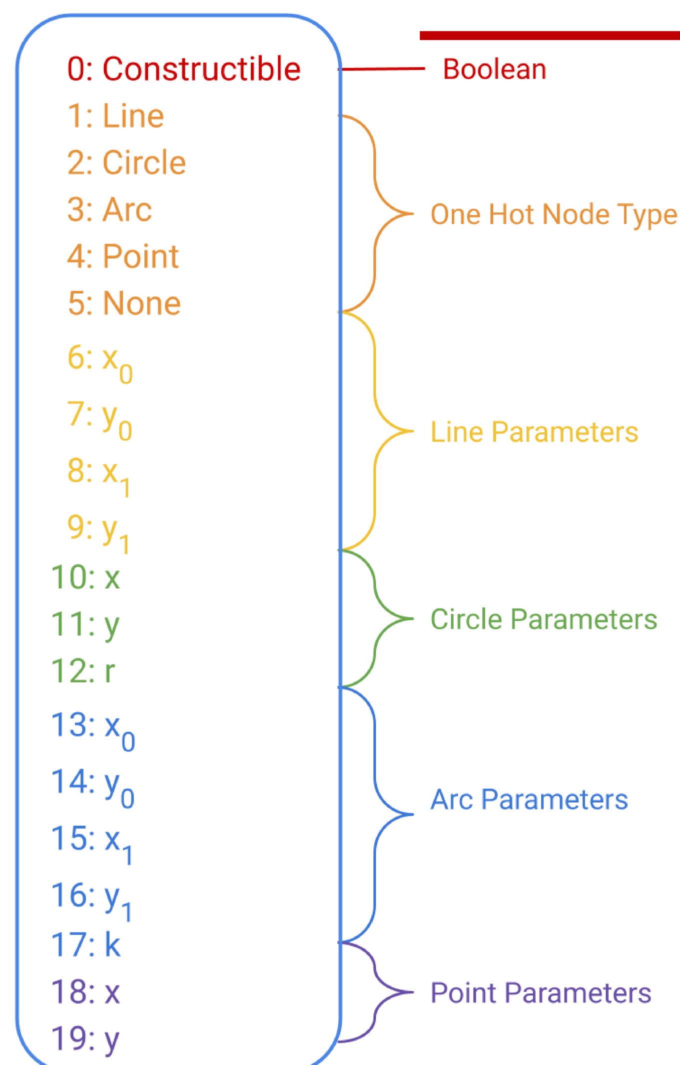


Fig 1. The blue line depicts the probability of the class label remaining unchanged, computed using Monte-Carlo estimation with 10,000 samples. **Left:** The cosine variance schedule is used directly. **Right:** The cosine schedule is augmented, resulting in information degrading more gradually.

Architecture



- **Heterogeneous Primitive Parameterizations**
  - We represent each primitive as a superposition (probabilistic mixture) of all primitive types.
  - Not only does this approach provide a generic representation of all primitives, but it also allows our model to explore all possible realizations of a primitive concurrently.
- **Permutation Invariant Denoising**
  - We employ the DiT architecture and simply omit positional encodings. Since, all the attention and feed-forward blocks are permutation equivariant, the model is as well.
  - Given the predicted noiseless sketch, each primitive is independently denoised with respect to its noiseless counterpart. This makes the denoising process invariant to the relative primitive orderings.

Training and Inference

For continuous variables ($\boldsymbol{x}$) we employ standard Gaussian diffusion, whereas for discrete variables ($\boldsymbol{y}$) we use Gaussian-Softmax diffusion. Accordingly, we employ MSE loss for parameters and CE loss for class labels.

Algorithm 1 Training Procedure

Require: Data with continuous and discrete information $(x_0 || y_0)$, Denoiser model $M_\theta(X)$, variance schedule $\bar{a}$, augmented variance schedule $\bar{b}$

1: **while** not converged **do**
2: Sample timestep $t \sim U(1, T)$
3: Add noise to parameters and labels $x_t || y_t = \text{forward}(x_0 || y_0, t)$

$$x_t || y_t \sim \mathcal{N}(\sqrt{\bar{a}_t} x_0, (1 - \bar{a}_t) I) \times \mathcal{GS}\left(\sqrt{\bar{b}_t} \log y_0, (1 - \bar{b}_t) I\right)$$

4: Reconstruct original sketch $(x' || y') = M_\theta(x_t || y_t, t)$
5: Mask out irrelevant parameters in x' according to true class label y_0

$$x' \leftarrow \text{mask}(x', y_0)$$

6: Compute reconstruction loss: $MSE(x', x_0) + CE(y', y_0)$
7: Update θ using gradient descent

Algorithm 2 Inference Procedure

Require: Denoiser model $M_\theta(\mathcal{V}, \mathcal{E})$, Random seed $x_T || y_T \sim \mathcal{N}(0, I) \times \mathcal{GS}(0, I)$

1: **for** $t = T - 1$ to 1 **do**
2: Predict noiseless datapoint $(x' || y') = M_\theta(x_t || y_t, t)$
3: Weight parameters in x' by corresponding label confidence in y', rescaled such that the maximum element is exactly 1

$$x' \leftarrow x' * y' / \max(y')$$

4: Interpolate noisy data with prediction according to the reverse transition

$$x_{t-1} || y_{t-1} = \text{reverse}(x_t || y_t, x' || y', t)$$

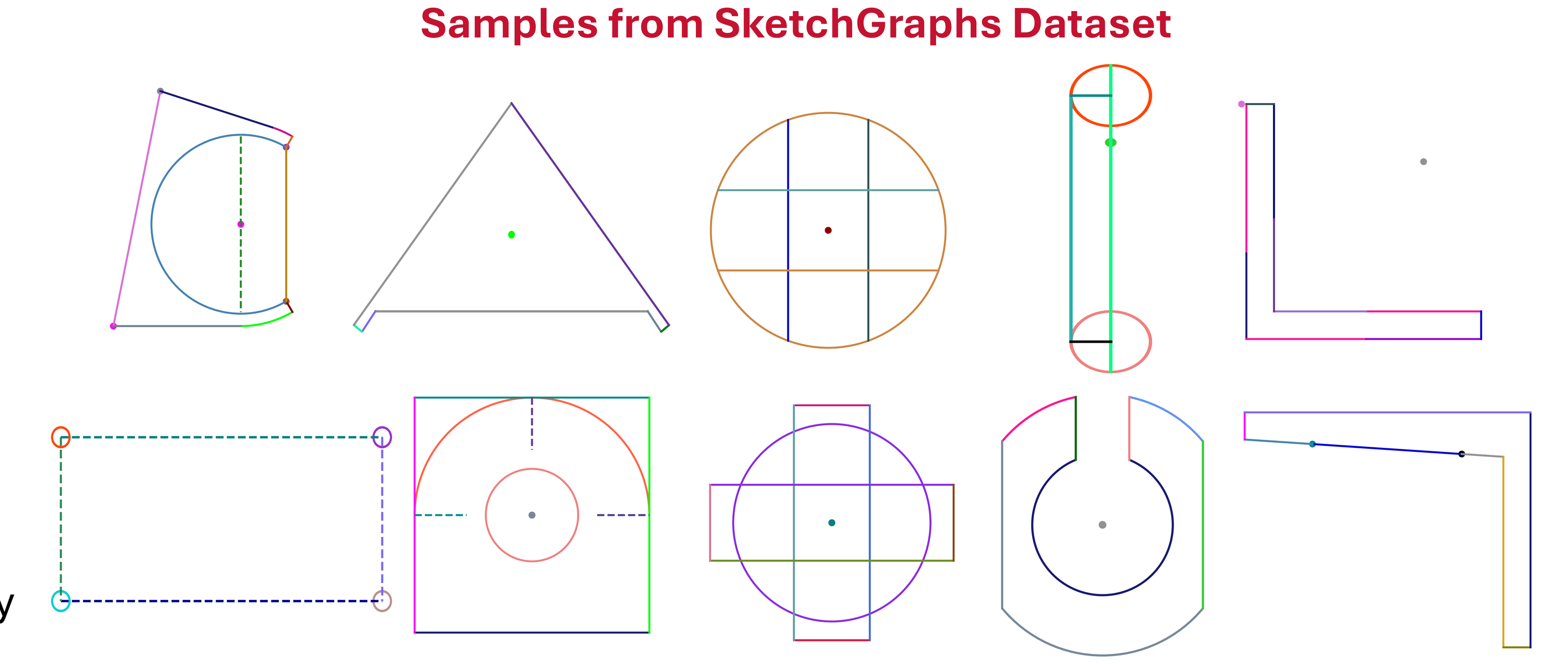
Quantitative Results

Method	Bits/Sketch↓	Bits/Primitive↓
SketchDNN (Ours)	**81.33**	**5.08**
SketchDNN (Pos.)	83.03	5.18
SketchDNN (Cat.)	106.10	6.63
Vitruvion	84.80	8.19
SketchGen	88.22	8.60

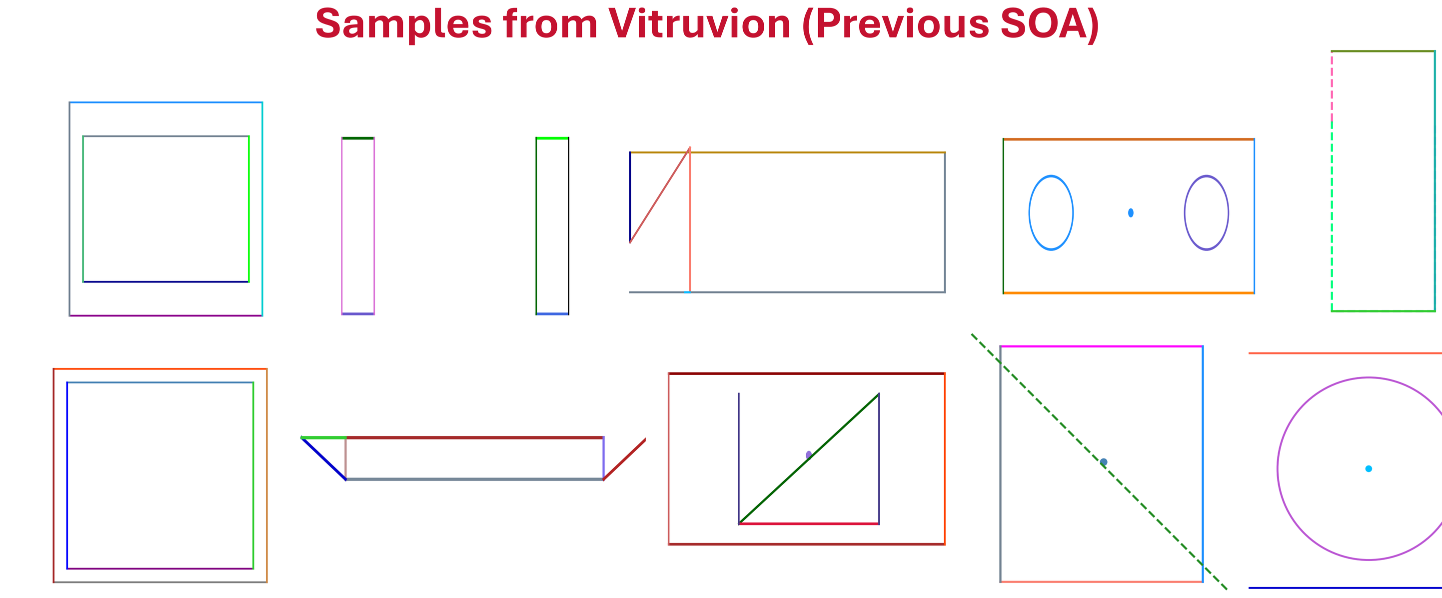
Method	FID 10K↓	Precision↑	Recall↑
SketchDNN (Ours)	**7.80**	0.246	**0.266**
SketchDNN (Pos)	10.26	0.230	0.245
Latent Diffusion	93.34	0.134	0.033
SketchDNN (Cat.)	148.93	0.117	0.028
Vitruvion	16.04	**0.294**	0.176

Qualitative Comparison

Samples from SketchGraphs Dataset



Samples from Vitruvion (Previous SOA)



Samples from SketchDNN (Ours)