Extended Abstract

Motivation SketchRNN generates vector sketches one stroke at a time by imitating human drawings from the Quick, Draw! dataset (Ha and Eck, 2017), with no objective tied to the quality of the final rendered image. The model captures stroke statistics well but has no awareness of whether the completed sketch is recognizable. We investigate whether reinforcement learning with image-based rewards can close this gap, improving semantic quality while preserving the diversity of the pretrained model.

Method We fine-tune the SketchRNN decoder using REINFORCE with the encoder frozen, comparing three reward functions: (1) an SSIM reward measuring pixel-level similarity to a fixed target owl image; (2) a sparse CLIP reward scoring the final rendered sketch against the prompt "a sketch of an owl" using CLIP ViT-B/32 (Radford et al., 2021); and (3) a dense CLIP reward computing CLIP scores every $k = 25$ strokes throughout the episode and summing them into a discounted return $G = \sum_t \gamma^{T-t} \cdot r_t$. To reduce gradient variance, batches of 16 episodes share a single latent vector z, isolating decoder stochasticity as the sole source of within-batch reward variance and enabling within-batch advantage normalization.

Implementation We built on the pretrained SketchRNN owl checkpoint from Google Magenta's Quick, Draw! model zoo. CLIP rewards use softmax confidence against four negative prompts (cat, dog, bird, fish). Dense rewards rasterize partial stroke sequences at each checkpoint before scoring with CLIP. All experiments train for 5,000 episodes with batch size 16, Adam at lr $= 10^{-5}$, gradient clipping to $[-1, 1]$, sampling temperature 0.5, and 3 warm-up batches. We also debugged a renderer issue present in our milestone experiments where the pen was incorrectly staying down between strokes, and re-ran all experiments with the fixed renderer. Training ran on Modal cloud GPUs (NVIDIA T4).

Results We evaluated 2,000 generated sketches per method across CLIP softmax confidence, FID vs. QuickDraw owls, precision and recall, and an independent ResNet18 QuickDraw classifier. Dense CLIP reward achieved the highest CLIP score (0.644 vs. 0.478 baseline, $+35\%$) with a 96.4% classifier rate, confirming gains were not CLIP-gaming. All RL methods increased FID relative to baseline (81.47), indicating a tradeoff between CLIP-optimized quality and similarity to the QuickDraw distribution. Sparse CLIP reward mode-collapsed (recall 0.156 vs. 0.254 baseline); dense reward partially mitigated this (recall 0.244). SSIM reward failed to generalize, producing the lowest CLIP score (0.284) and classifier rate (86.0%).

Discussion Dense rewards provided a meaningful improvement over sparse rewards, likely by giving early strokes better credit assignment. The FID increase across all RL methods suggests that CLIP optimization pushes sketches toward a visual style CLIP finds more owl-like, but farther from human Quick Draw sketches — an expected tradeoff. The recall gap between sparse and dense CLIP reward (0.156 vs. 0.244) suggests denser feedback during generation helps preserve diversity. One surprising result was that SSIM preserved more diversity than CLIP at the milestone stage, suggesting semantic reward optimization can cause stronger mode collapse than pixel-level similarity optimization.

Conclusion SketchRL shows that RL with image-based rewards can meaningfully improve the semantic quality of generative sketch models beyond stroke-level imitation learning. Dense intermediate rewards outperform sparse terminal rewards on semantic quality while better preserving diversity. Independent classifier evaluation confirms these gains reflect genuine improvement rather than reward hacking.

SketchRL: Finetuning Generative Sketch Models with Visual Rewards

Mallika Parulekar
Department of Computer Science
Stanford University
mallika.parulekar@stanford.edu

Hannah Levin
Department of Computer Science
Stanford University
levinh@stanford.edu

Tia Geri
Department of Computer Science
Stanford University
tgeri@stanford.edu

Abstract

SketchRNN generates vector sketches by imitating human drawings stroke by stroke, with no objective tied to the quality of the final rendered image. We introduce SketchRL, a REINFORCE-based framework that fine-tunes a pretrained SketchRNN decoder using visual rewards. We compare three reward functions – SSIM similarity to a target image, sparse CLIP reward at episode end, and dense CLIP reward computed at intermediate stroke checkpoints – evaluating across CLIP confidence, FID, precision, recall, and an independent QuickDraw classifier. Dense CLIP reward achieves the highest semantic quality (+35% CLIP score over baseline) while better preserving diversity than sparse reward. Independent classifier evaluation confirms gains are not due to reward hacking.

1 Introduction

SketchRNN (Ha and Eck, 2018) generates vector sketches one stroke at a time, trained on the Quick, Draw! dataset (Ha and Eck, 2017) via stroke-level supervision. While it captures how humans draw, it has no mechanism to optimize whether the final image is recognizable. The model often fails to lift the pen correctly and produces drawings that are messy or hard to identify, even though individual strokes are statistically plausible.

Reinforcement learning offers a natural solution: treat sketch generation as a sequential decision process, render the final image, and compute a reward that measures quality. This approach has been applied to image generation (Black et al., 2024) and language models (Ouyang et al., 2022), but remains underexplored for vector sketch generation.

We introduce **SketchRL**, which fine tunes a pretrained SketchRNN decoder using REINFORCE with visual rewards computed from rendered sketches. We compare SSIM, sparse CLIP, and dense CLIP reward functions and evaluate across five metrics including an independent classifier to detect reward hacking. Our main contributions are:

- A batched REINFORCE pipeline for SketchRNN with frozen encoder, shared latent z per batch, and within-batch advantage normalization.
- Comparison of three reward functions: SSIM, sparse CLIP, and dense CLIP with intermediate checkpoints every $k = 25$ strokes.
- Comprehensive quantitative evaluation across five metrics on 2,000 generated sketches per method.

- Evidence that dense intermediate rewards outperform sparse terminal rewards in semantic quality while better preserving diversity.

2 Related Work and Dataset

Quick, Draw! Dataset: The Quick, Draw! dataset (Ha and Eck, 2017) contains 50 million human sketches across 345 categories as vector stroke sequences, and is the standard benchmark for sketch generation.

SketchRNN: Ha and Eck (Ha and Eck, 2018) introduced SketchRNN, a sequence-to-sequence VAE that encodes sketches into a latent space and decodes them autoregressively as stroke sequences in stroke-5 format. Trained on Quick, Draw!, it captures stroke statistics but has no final-image objective.

CLIP: Radford et al. (Radford et al., 2021) introduced CLIP, a vision-language model trained on image-text pairs via contrastive learning. We use CLIP ViT-B/32 as a reward signal, scoring generated sketches against text prompts. CLIP has been used as a reward for image generation in (Black et al., 2024) but not, to our knowledge, for stroke-level generative models.

SSIM: The Structural Similarity Index (Wang et al., 2004) measures perceived image quality by comparing luminance, contrast, and structure between two images. We use it as an alternative reward to CLIP, optimizing directly for pixel-level similarity to a fixed target sketch.

REINFORCE: Williams (1992) introduced the REINFORCE policy gradient algorithm. We use a batched variant with within-batch advantage normalization to reduce gradient variance, similar to approaches used in LLM fine-tuning with human feedback Ouyang et al. (2022).

3 Method

3.1 SketchRNN Background

SketchRNN represents sketches in stroke-5 format: each token is $(\Delta x, \Delta y, p_1, p_2, p_3)$ where $(\Delta x, \Delta y)$ is the pen displacement and (p_1, p_2, p_3) are one-hot pen states (pen down, pen up, end of sketch). A bidirectional RNN encoder maps an input sketch to a Gaussian latent distribution; a decoder RNN samples from this distribution autoregressively to generate new strokes. At inference time, a latent vector z is sampled and decoded at a given temperature.

3.2 REINFORCE Pipeline

We fine-tune the decoder with REINFORCE, keeping the encoder frozen to preserve the pretrained latent space and reduce the number of trainable parameters. For each training batch, we sample a single latent vector z and generate 16 independent sketches from the decoder. This isolates decoder stochasticity as the sole source of within-batch reward variance, enabling cleaner gradient estimates.

For each episode i in the batch, we compute a reward r_i (see Section 3.3), normalize advantages within the batch:

$$\hat{A}_i = \frac{r_i - \bar{r}}{\sigma_r + \epsilon} \tag{1}$$

and compute a weighted average of per-episode gradients:

$$\nabla_\theta J \approx \frac{1}{B} \sum_{i=1}^{B} \hat{A}_i \cdot \nabla_\theta \log p_\theta(\tau_i) \tag{2}$$

Gradients are clipped to $[-1, 1]$ and applied with a single Adam update (lr $= 10^{-5}$). We use 3 warm-up batches with no weight updates to establish a stable reward baseline before training begins.

3.3 Reward Functions

SSIM Reward (sparse): At the end of each episode, the stroke sequence is rasterized to a 224×224 grayscale image and compared to a fixed target owl image using SSIM Wang et al. (2004). This directly optimizes pixel-level similarity to one specific reference sketch.

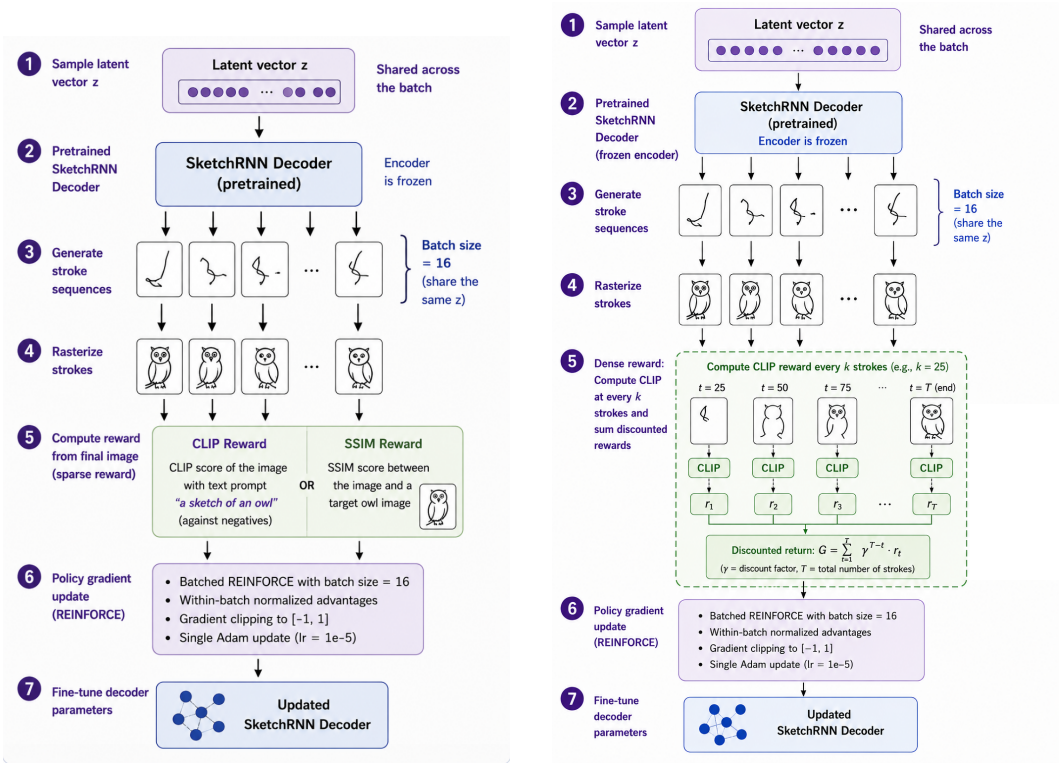

Figure 1: RL pipelines for sparse rewards (left) and dense rewards (right). Both freeze the SketchRNN encoder and fine-tune only the decoder. The dense pipeline computes CLIP rewards at intermediate stroke checkpoints every $k = 25$ strokes, summing into a discounted return $G = \sum_t \gamma^{T-t} \cdot r_t$. We use $\gamma = 1$ (undiscounted) – this gives every stroke equal credit, directly addressing the credit assignment problem from sparse reward methods that optimized for later strokes over early partial drawings.

CLIP Reward (sparse): At the end of each episode, the completed sketch is rasterized to a 224×224 RGB image and scored with CLIP ViT-B/32 Radford et al. (2021). The reward is the softmax probability assigned to the positive prompt "a sketch of an owl" against four negative prompts ("a sketch of a cat", "a sketch of a dog", "a sketch of a bird", "a sketch of a fish").

CLIP Reward (dense): Rather than scoring only the final image, we compute CLIP rewards at intermediate checkpoints throughout the episode every $k = 25$ strokes. At each checkpoint t, we rasterize the partial stroke sequence and compute the CLIP softmax score. The episode return is:

$$G = \sum_{t=1}^{T} \gamma^{T-t} \cdot r_t \tag{3}$$

where T is the total number of strokes and γ is a discount factor. With $\gamma = 1.0$ (our default), this reduces to a simple sum of all intermediate rewards. Setting $\gamma < 1$ up-weights later checkpoints, which correspond to more complete sketches and thus more reliable quality signals. This formulation provides reward feedback throughout the episode rather than only at the end, improving credit assignment for early strokes.

4 Experimental Setup

Model: We use the pretrained SketchRNN owl checkpoint from Google Magenta's Quick, Draw! model zoo, with the encoder frozen throughout training.

Training: All experiments run for 5,000 episodes (312 batches) with batch size 16, sampling temperature 0.5, maximum sequence length 250 stroke tokens, and 3 warm-up batches. We train on Modal cloud GPUs (NVIDIA T4).

Evaluation: We generate 2,000 sketches per method at temperature 0.5 and evaluate on five metrics:

- **CLIP score**: softmax confidence of "a sketch of an owl" against the same four negative prompts used during training.
- **FID**: computed from InceptionV3 pool features extracted from generated and reference images using scipy for matrix square root.
- **Precision and Recall**: computed using the `prdc` library (Naeem et al., 2020) from InceptionV3 features with nearest $k = 5$, measuring realism and diversity respectively.
- **QuickDraw Classifier**: a ResNet18 fine-tuned on QuickDraw sketch categories (owl, cat, dog, bird, fish), used as an independent measure of owl recognizability that is not influenced by the CLIP reward signal.

5 Results

5.1 Quantitative Evaluation

Method	CLIP↑	FID↓	Precision↑	Recall↑	Classifier↑
Baseline	0.478	81.47	0.475	0.254	97.7%
CLIP (sparse)	0.585	89.24	0.515	0.156	99.1%
SSIM (sparse)	0.284	90.96	0.428	0.187	86.0%
CLIP (dense, λ = 1)	**0.644**	89.71	0.465	0.244	96.4%

Table 1: Evaluation across 2,000 generated owl sketches per method (n=2,000). CLIP = softmax confidence vs. negative prompts; FID = Fréchet Inception Distance vs. QuickDraw owls; Classifier = independent ResNet18 owl recognition rate.

Method	CLIP↑	FID↓	Precision↑	Recall↑	Classifier↑
CLIP dense γ=1.0	0.389	116.09	0.420	0.259	76.8%
CLIP dense γ=0.95	0.678	**89.59**	**0.516**	0.205	99.2%
CLIP dense γ=0.9	0.687	92.54	0.478	0.231	98.4%
CLIP dense γ=0.75	0.662	92.71	0.484	0.229	98.6%
CLIP dense γ=0.5	0.688	93.37	0.446	**0.238**	**99.4%**
CLIP dense γ=0.25	**0.702**	96.55	0.476	0.198	97.8%

Table 2: Discount factor ablation across n=500 generated owl sketches.

Dense CLIP reward achieved the highest CLIP score (0.644 vs. 0.478 baseline, $+35\%$ relative improvement). Interestingly, the independent classifier agreed with CLIP rankings — dense reward scored 96.4%, well above chance and only slightly below baseline (97.7%) — confirming that gains reflect genuine improvement in owl recognizability rather than gaming the CLIP reward signal.

All RL methods increased FID relative to baseline (81.47), which was expected: CLIP optimization pushes sketches toward a style that CLIP finds more owl-like, but further from the Quick Draw human drawing distribution. This is a known tension in reward-based fine-tuning.

Sparse CLIP reward mode-collapsed: recall dropped to 0.156 vs. 0.254 baseline, while precision was highest (0.515). The model converged to a narrow set of recognizable owl sketches, sacrificing diversity. Dense reward partially mitigated this, preserving recall at 0.244 — suggesting that reward feedback throughout the episode helps the model maintain a broader set of drawing strategies.

SSIM reward failed to generalize. Optimizing pixel similarity to a single target image yielded the lowest CLIP score (0.284) and classifier rate (86.0%), with roughly 1 in 8 generated sketches classified as non-owl. The model appeared to mimic structural features of the target image rather than learning the concept of an owl, and the fixed reference provided no signal for semantic generalization.

We explore discount factor γ across six values to study its effect on dense rewards. Across n=500 generated owl sketches, CLIP dense with $\gamma = 0.25$ achieved the highest CLIP score but at the cost of the lowest recall (0.198). Similarly, the second lowest γ (0.5), achieved the highest independent ResNet 18 owl recognition rate. With a lower γ, the model puts less weight on early strokes and more attention to the final version of the drawing, thus memorizing the final style more and repeating it rather than encouraging diversity.

5.2 Qualitative Analysis

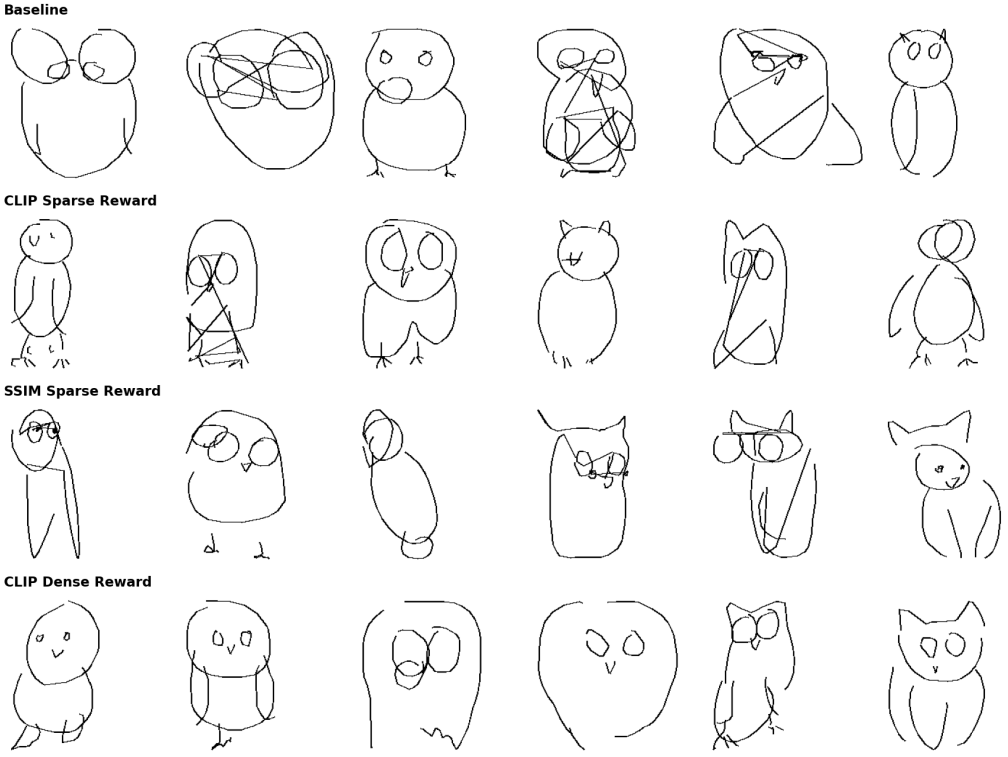

Figure 2: Generated owl sketches from each method. **Baseline:** Varied styles but messy, often failing to lift the pen when needed between strokes. **CLIP Sparse Reward:** Cleaner drawings but lost diversity of style. **SSIM Sparse Reward:** Diverse but unrecognizable with chaotic body shapes. **CLIP Dense Reward:** Cleanest and most diverse drawings

Qualitative results in Figure 2 are consistent with the quantitative findings. Sparse CLIP sketches are recognizable owls but converge on similar poses and structures. SSIM sketches show the head tilt and body shape of the target image but lack owl-specific features like eyes or feathers that differ from the reference. Dense CLIP sketches are the most consistently recognizable while showing more variety in pose and detail than sparse CLIP.

6 Discussion

Dense rewards outperformed sparse rewards on both CLIP score and diversity, which we attribute to better credit assignment. With sparse rewards, early strokes receive weak gradient signal because the reward is only observed after the full sketch is drawn. Dense rewards provide feedback at each 25-stroke checkpoint, giving the model more direct signal about which early stroke decisions lead to more recognizable partial drawings.

The FID increase across all RL methods is an expected tradeoff: QuickDraw owls are quick, messy human sketches, while CLIP-optimized owls are cleaner and more structured. These are different

visual styles, and FID measures distance from the human drawing distribution rather than absolute quality.

7 Conclusion

SketchRL demonstrates that RL with image-based rewards can meaningfully improve the semantic quality of a generative sketch model beyond stroke-level imitation learning. Dense intermediate rewards achieve the best CLIP score (+35% over baseline) while better preserving diversity than sparse rewards, and independent classifier evaluation confirms these gains reflect real improvement rather than reward hacking. All RL methods trade off distributional similarity to QuickDraw for improved CLIP recognizability, which is an expected consequence of optimizing for a different objective than the pretraining signal.

Future work could explore alternative RL algorithms such as PPO or Actor-Critic to reduce variance, and run experiments to other Quick Draw categories beyond owls to test generalization.

8 Team Contributions

- **Mallika Parulekar:** Implemented the SSIM-based RL fine-tuning pipeline, debugged and fixed the stroke renderer, ran SSIM reward experiments, and contributed to poster and report writing.
- **Hannah Levin:** Set up the SketchRNN training infrastructure, implemented the evaluation pipeline and dense reward RL fine-tuning pipeline, ran evaluation across all experiments, and contributed to poster and report writing.
- **Tia Geri:** Implemented the sparse CLIP-based RL fine-tuning pipeline, designed the batched REINFORCE training protocol, ran CLIP reward experiments, and contributed to poster and report writing.

Changes from Proposal We pivoted to using the SketchRNN framework as a starting point from which to experiment with different reinforcement learning methods to improve the sketches, instead of implementing the sketching system from scratch.

References

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. 2024. Training Diffusion Models with Reinforcement Learning. In *International Conference on Learning Representations*. https://arxiv.org/abs/2305.13301

David Ha and Douglas Eck. 2017. Quick, Draw! Dataset. https://github.com/googlecreativelab/quickdraw-dataset. Accessed: 2026-05-22.

David Ha and Douglas Eck. 2018. A Neural Representation of Sketch Drawings. *International Conference on Learning Representations* (2018). https://arxiv.org/abs/1704.03477

Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. 2020. Reliable Fidelity and Diversity Metrics for Generative Models. In *International Conference on Machine Learning*. PMLR, 7176–7185.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training Language Models to Follow Instructions with Human Feedback. *Advances in Neural Information Processing Systems* 35 (2022), 27730–27744.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning (ICML)*.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. https://www.cns.nyu.edu/pub/eero/wang03-reprint.pdf. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612. Accessed: 2026-05-23.

Ronald J. Williams. 1992. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning* 8, 3 (1992), 229–256.