gan-rl-report

Bardia Rezaeimehr

December 2025

1 Generative Adversarial Network (GAN)

1.1 Dataset and Preprocessing

The CartoonSet dataset consists of cartoon face images with associated metadata describing facial attributes. For the GAN baseline, only the image data is used. All images are resized to a fixed resolution and normalized to the range $[-1, 1]$ to match the output range of the generator.

After experimenting with different resolutions, the final GAN is trained using images of size 64×64. This resolution provided significantly improved training stability and diversity compared to higher resolutions (e.g., 128×128), while remaining sufficient for capturing global facial structure.

1.2 Generator Architecture

The generator is implemented as a Deep Convolutional GAN (DCGAN)-style architecture. It takes as input a latent vector $z \in R^{128}$ and progressively upsamples it through a sequence of transposed convolution layers.

The generator consists of:

- A fully convolutional architecture starting from a 4×4 feature map.
- A series of transposed convolution layers with stride 2 to increase spatial resolution.
- Batch Normalization and ReLU activations after each intermediate layer.
- A final `tanh` activation to produce RGB images in the range $[-1, 1]$.

This design allows the generator to gradually learn global structure before refining local details.

1.3 Discriminator Architecture

The discriminator is a convolutional neural network that maps an input image to a single scalar logit. It progressively downsamples the image using strided convolution layers.

Key design choices include:

- LeakyReLU activations to avoid dead neurons.
- Spectral Normalization applied to all convolution layers to enforce Lipschitz continuity and stabilize training.
- No Batch Normalization, as spectral normalization already constrains the discriminator.

The final layer outputs a single logit value representing the discriminator's confidence.

1.4 Training Procedure

The generator and discriminator are trained alternately using mini-batch stochastic gradient descent. Binary Cross-Entropy loss is used for both networks, with real images labeled as 1 and fake images labeled as 0.

The Adam optimizer is employed with carefully tuned learning rates to balance the adversarial game. Training stability was monitored through loss curves and visual inspection of generated samples.

Several stabilization techniques were explored during development, including learning rate tuning, update frequency balancing, and resolution reduction. The final configuration achieved stable convergence without mode collapse.

1.5 Results and Observations

The trained GAN successfully generates diverse and visually coherent cartoon faces. At the chosen resolution of 64×64, the model captures key facial attributes such as hair style, skin tone, and face shape.

While random sampling from the latent space already produces reasonable diversity, some samples exhibit artifacts or incomplete features. This observation motivates the reinforcement learning component, which aims to guide latent selection toward regions of the latent space associated with higher-quality outputs.

2 Reinforcement Learning for Latent Selection

2.1 RL Formulation

The RL problem is formulated as a discrete-action, single-state Markov Decision Process, equivalent to a multi-armed bandit setting. The components are defined as follows:

- **State:** A single, constant state. Since the GAN is frozen and the environment does not change, no explicit state representation is required.
- **Action:** Selecting an index $a \in \{1, \dots, K\}$ corresponding to one of K predefined latent vectors stored in a latent codebook.

- **Reward:** A scalar value derived from the discriminator output. For a selected latent vector z_a, an image $x = G(z_a)$ is generated and passed through the discriminator. The reward is defined as:

$$r = \sigma(D(G(z_a))) \tag{1}$$

  where $\sigma(\cdot)$ denotes the sigmoid function.

- **Policy:** An ϵ-greedy policy over the latent indices, balancing exploration and exploitation.

- **Value Function:** A table $Q(a)$ that estimates the expected reward of selecting latent index a.

2.2 RL Algorithm

The RL agent maintains a value estimate $Q(a)$ for each latent vector in the codebook. At each iteration, an action is selected using an ϵ-greedy strategy. After observing the reward, the value estimate is updated using an exponential moving average:

$$Q(a) \leftarrow (1 - \alpha)Q(a) + \alpha r \tag{2}$$

where α is the learning rate. Since the environment has no state transitions, this update rule corresponds to a bandit-style value learning algorithm.

The exploration rate ϵ is gradually annealed during training, allowing the agent to explore broadly in early stages and exploit high-reward latent vectors once learning has converged.

2.3 Training Setup

During RL training, both the generator and discriminator are frozen and used only for inference. This ensures that the RL agent does not alter the learned data distribution, but instead learns a sampling strategy over the existing latent space.

The latent codebook is initialized by sampling K latent vectors from a standard normal distribution. The agent is trained for several thousand iterations until the reward moving average and policy entropy stabilize, indicating convergence.

2.4 Evaluation and Results

The performance of the RL agent is evaluated by comparing images generated from:

- Randomly sampled latent vectors, and
- Latent vectors with the highest learned Q values.

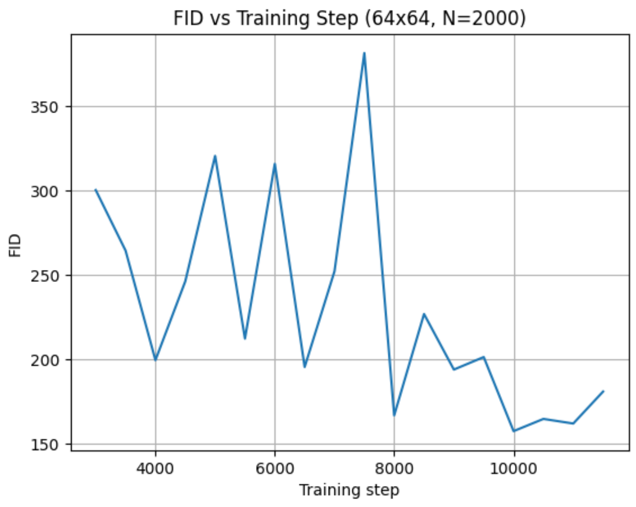

Figure 1: FID versus training step for the 64×64 GAN (Inception-v3, $N = 2000$). Lower is better.

Qualitative results demonstrate that RL-selected latent vectors consistently produce more complete, visually coherent, and confident facial representations. Attributes such as hair style, facial symmetry, and accessories (e.g., sunglasses) appear more clearly in RL-guided samples. While random sampling exhibits higher diversity, it also produces a larger proportion of low-quality samples.

The learned policy maintains non-zero entropy, indicating that the agent does not collapse to a single latent vector, but instead distributes probability mass over multiple high-quality regions of the latent space.

3 Evaluation

3.1 Visual Quality and Fréchet Inception Distance (FID)

The resulting FID values exhibit an overall decreasing trend as training progresses, indicating improvement in the visual quality and statistical similarity of generated images to the real data distribution.

Specifically, FID decreases from approximately 300 at training step 3000 to a minimum value of 157.44 at step 10000. While temporary increases in FID are observed at certain intermediate checkpoints, such fluctuations are expected in adversarial training due to the non-stationary dynamics of GAN optimization and the finite-sample estimation of FID. Overall, the downward trend confirms that the generator progressively learns a better approximation of the real data distribution.

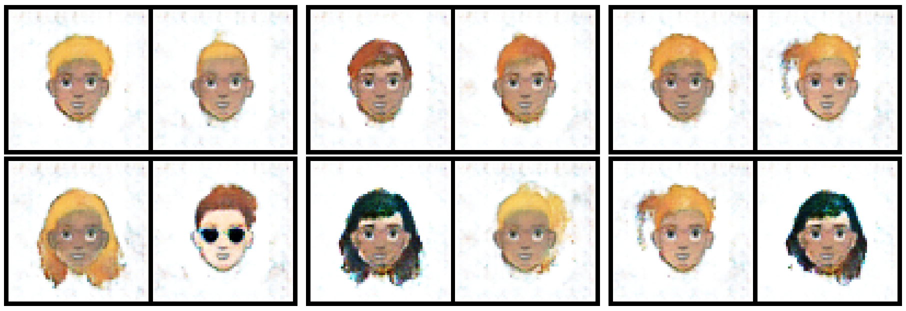

Figure 2: Generated-image pairs with low LPIPS (top: visually similar) and high LPIPS (bottom: visually diverse) from the step 11500 checkpoint.

3.2 Diversity of Generated Images and LPIPS

LPIPS was computed on 100 randomly selected pairs of images generated by the GAN using random latent vectors from the checkpoint at training step 11500. The generator outputs were directly used in the range $[-1, 1]$, as required by the LPIPS formulation. The resulting statistics are as follows:

- Mean LPIPS: 0.173
- Standard deviation: 0.063
- Minimum LPIPS: 0.051
- Maximum LPIPS: 0.350

3.2.1 Low and High LPIPS Pair Visualization

We visualize at least three pairs with low LPIPS (high similarity) and three pairs with high LPIPS (high diversity). The results can be seen in Figure 2.

Overall, the LPIPS evaluation confirms that the trained GAN produces a diverse set of images while preserving visual coherence, satisfying the diversity requirements of the assignment.

3.3 Discriminator Performance

3.3.1 True/False Positive/Negative Examples

To qualitatively analyze discriminator behavior, we include example images for each case: True Positive (fake $\rightarrow$ fake), False Positive (real $\rightarrow$ fake), True Negative (real $\rightarrow$ real), and False Negative (fake $\rightarrow$ real). The results can be seen in Figure 3.

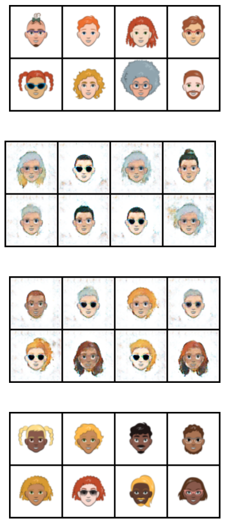

Figure 3: Example images for each discriminator outcome at step 11500 (From top to bottom): TP (fake → fake), FP (real → fake), TN (real → real), FN (fake → real).

3.3.2 Discriminator's loss Diagram

The discriminator loss remains bounded within a reasonable range and does not diverge or collapse to zero. This suggests that the discriminator neither becomes trivially perfect nor completely ineffective. Such behavior is a sign of stable GAN training, especially when combined with spectral normalization, which helps prevent excessive discriminator confidence and gradient explosion. Loss logs can be seen in Figure 4

3.3.3 Confusion Matrix Analysis

The confusion matrix provides insight into the discriminator's ability to distinguish real and generated images. As it can be seen in Figure 5, majority of real images are correctly classified as real (True Positives), and most generated images are correctly identified as fake (True Negatives), indicating that the discriminator has learned meaningful decision boundaries.

3.4 Training Stability

The observed loss curves in both Discriminator and Generator demonstrate stable adversarial training after appropriate tuning of learning rates, update frequencies, and image resolution. No mode collapse or divergence was observed in the final configuration

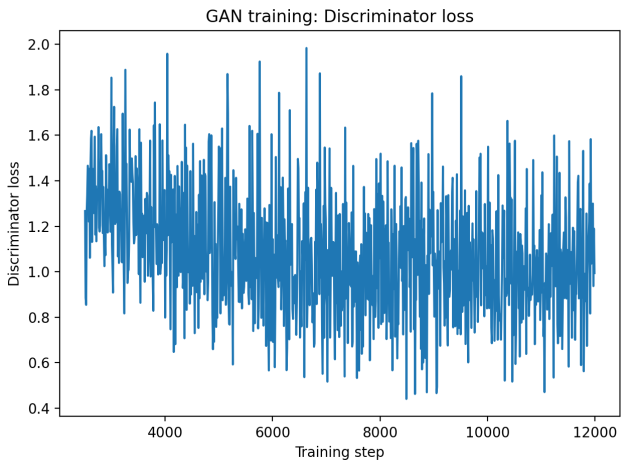

Figure 4: Discriminator loss throughout training.

3.5 Reproducibility with Fixed Random Seed

To verify reproducibility, the generator was evaluated using a fixed random seed. Generated images produced from identical latent vectors across multiple runs remain visually consistent. This can be observed in Figure 7.

3.6 RL Agent Evaluation and Policy Entropy

At the beginning of training, entropy rapidly increases, indicating strong exploration as the agent samples a wide range of latent actions. This phase allows the agent to discover diverse regions of the GAN latent space.

After reaching a peak, entropy gradually decreases over time, reflecting a transition from exploration to exploitation. This entropy behavior aligns with the improvement in reward moving average and demonstrates that the RL agent successfully balances exploration and exploitation while learning to guide the GAN toward higher-quality outputs. The entropy curve is shown in Figure.8

3.7 Summary

Overall, the evaluation results demonstrate that the proposed GAN achieves satisfactory visual quality, diversity, and training stability. Furthermore, the RL agent effectively guides latent selection toward higher-quality samples while maintaining behavioral diversity. These findings validate the effectiveness of the proposed GAN–RL framework.

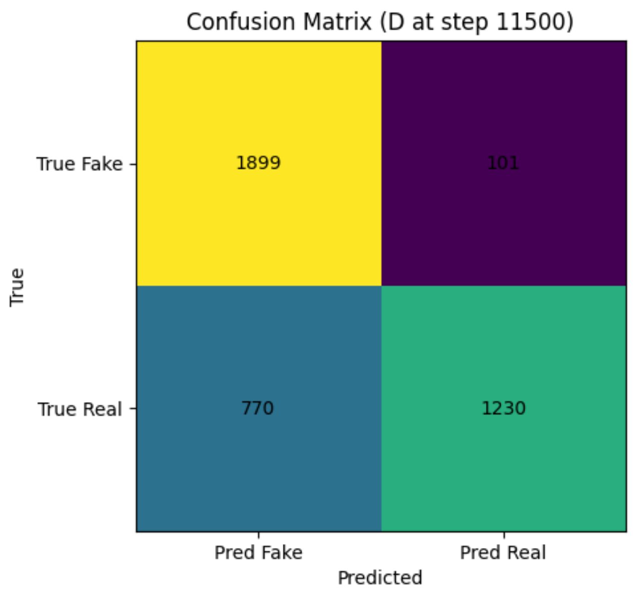

Figure 5: Confusion Matrix

4 Theoretical Questions

This section addresses the theoretical questions related to Generative Adversarial Networks (GANs) and their integration with Reinforcement Learning (RL). The discussion connects theoretical concepts with observations from the implemented models and experimental results.

4.1 Question 1

If the discriminator becomes excessively strong, it can perfectly distinguish real samples from generated ones. In this case, the discriminator outputs saturate, leading to near-zero gradients for the generator. As a result, the generator receives insufficient learning signals and its training becomes unstable or stalls completely.

4.2 Question 2

This balance emerges through simultaneous adversarial optimization, where each network continuously adapts to the other. When the balance is disrupted—either by an overly strong discriminator or an undertrained one—the adversarial game fails, resulting in poor image quality.

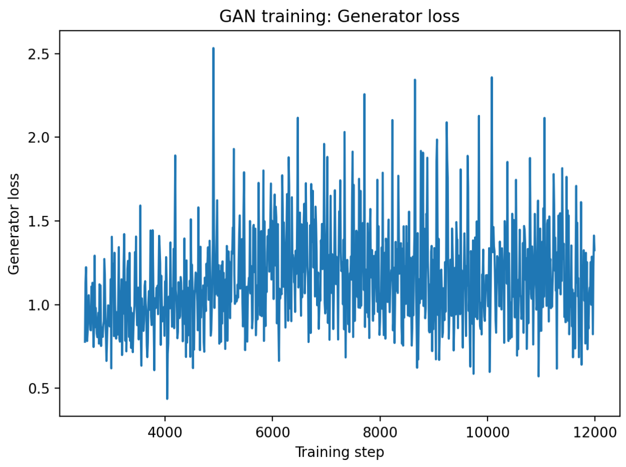

Figure 6: Generator loss throughout training.

4.3 Question 3

A commonly used choice for GAN training is the non-saturating loss. The discriminator minimizes a binary classification loss to distinguish real and fake samples, while the generator maximizes the discriminator's likelihood of classifying generated images as real. This loss formulation improves gradient flow to the generator and leads to more stable training.

4.4 Question 4

Excessive normalization or aggressive preprocessing may remove important structural or textural information from images. This can unintentionally lead to a stronger discriminator, as real images become easier to separate from generated ones.

4.5 Question 5

For example, rewards can be increased for generated images that exhibit rare or underrepresented attributes according to metadata statistics. This approach guides the agent toward selecting latent vectors that promote diversity or target specific visual features.

4.6 Question 6

In Q-learning, excessively high learning rates can cause unstable updates. Large updates may overshoot optimal Q-values, leading to oscillations or divergence. This instability prevents convergence of the value function and results in unpredictable agent behavior, especially in noisy or non-stationary environments.

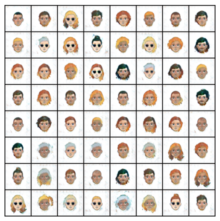
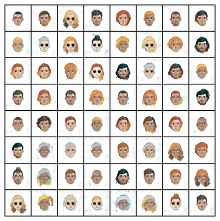

Figure 7: Generated images using fixed random seed.

4.7 Question 7

Reward shaping can accelerate learning by providing dense and informative feedback. On the other hand, poorly designed reward functions may cause unintended biases, encouraging the agent to exploit shortcuts that maximize reward without improving quality or diversity.

4.8 Question 8

One efficient way is that rewards can favor balanced attribute distributions or penalize overrepresented features, enabling controlled generation while preserving visual realism.

4.9 Question 9

Multiple agents with different reward objectives can be combined using strategies such as weighted reward aggregation, hierarchical agents, or alternating training phases. These methods allow agents to encourage different stylistic or diversity goals without sacrificing overall image quality.

4.10 Question 10

Discrete latent selection simplifies the agent's action space but limits expressiveness, while higher-dimensional latent spaces increase diversity at the cost of more complex exploration.

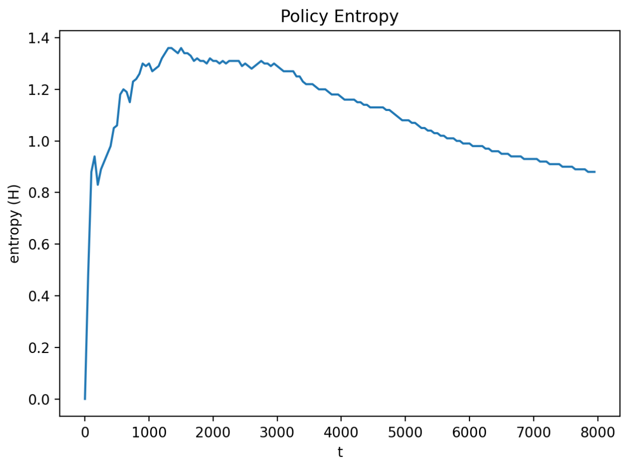

Figure 8: RL Entropy Curve.

4.11 Question 11

Methods like cumulative reward trends, policy entropy, FID and LPIPS scores, and visual inspection of generated samples can be applied.

4.12 Question 12

Multiple agents can be combined using shared replay buffers or hierarchical control.

4.13 Question 13

The reinforcement learning components are defined as follows:

- **State**: Current latent index or latent-space statistics
- **Action**: Selection of a latent vector (or latent index)
- **Reward**: Discriminator-based score or shaped reward
- **Policy**: Probability distribution over latent selections
- **Value Function**: Expected cumulative reward for a given state