Neon: Negative Extrapolation from Self-training improves Image Generation (ICLR 2026 Oral)

presented by Choeun Kim

March 5, 2026

IDEA, Department of Statistics, Seoul National University

Table of Contents

Introduction

Motivation: The Data Bottleneck & MAD

- **Data Bottleneck**: Scaling generative AI models is heavily bottlenecked by the scarcity of high-quality training data.
- **Naïve Self-training**: Using unverified synthetic data to augment real data seems promising, but leads to a positive feedback loop.
- **Model Autophagy Disorder (MAD)**: This loop causes a phenomenon known as model collapse, resulting in rapid degradation of sample quality and diversity.
- **Limitations of Existing Solutions**: Workarounds like external verifiers or auxiliary discriminators add significant computational overhead and architectural constraints.

Proposed Method: Neon

Notation

- $\mathcal{D}$: a training data set drawn from p_{data}
- G_θ : the generative model
- $\mathcal{B}$: training budget, the cumulative number of images seen (global steps)$\times$(global batch size)
- $\mathcal{I}$: inference routine
- $q_{\theta,\kappa}$: sampling distribution induced with $\mathcal{I}$ and hyperparameters κ

  e.g., diffusion reverse process with cfg (guidance scale κ)
- p_θ : sampling distribution without inference-time modifications ($p_\theta := q_{\theta,\varnothing}$)
- $|\cdot|$: set cardinality
- $\|x\|_M := \|M^{1/2}x\|_2, \langle x, y\rangle_M := x^T My$

  $\|\cdot\|_2, \langle\cdot,\cdot\rangle$: Euclidean norm, inner product

Neon: Negative Extrapolation from Self-traiNing

Key Insight

The degradation from self-training is not random noise, but a powerful signal that is anti-aligned with the real-data population gradient.

Instead of avoiding degradation, Neon turns it into a signal for self-improvement.

- **Step 1**: Fine-tune a base model (θ_r) on its own self-synthesized data to get a degraded model (θ_s).
- **Step 2**: Reverses the gradient updates to extrapolate *away* from the degraded weights.

$$\theta_{Neon} = (1 + w)\theta_r - w\theta_s \quad (w > 0)$$

- w controls the extrapolation strength.

Algorithm: Neon

Algorithm 1 Neon: Negative Extrapolation from Self-Training

Require: Base model G_{θ_r}, inference routine $\mathcal{I}$ with hyperparameters κ

1: **Hyperparameters:** Synthetic dataset size $n_s = |\mathcal{S}|$, extrapolation strength w, training budget $\mathcal{B}$

2: $\mathcal{S} \leftarrow \{x_i\}_{i=1}^{n_s}$ where $x_i \sim q_{\theta_{r,\kappa}}$ induced by $\mathcal{I}(G_{\theta_r}; \kappa)$ ▷ sample using test-time inference

3: $G_{\theta_s} \leftarrow \text{FineTune}(G_{\theta_r}, \mathcal{S}, \mathcal{B})$ ▷ briefly fine-tune on synthetic data

4: $\theta_{Neon} \leftarrow (1 + w)\theta_r - w\theta_s$ ▷ reverse the degradation

output Final generator $G_{\theta_{Neon}}$

- Deceptively simple post-hoc merge.
- Requires no new real data, no auxiliary models, and no inference modifications.

Theoretical Analysis

Why Neon Works: Set-up

- $l_\theta(x)$: differentiable (w.r.t. θ) loss function
- $\mathcal{R}_{\text{data}}(\theta) := \mathbb{E}_{p_{\text{data}}}[l_\theta(X)]$, $\mathcal{R}_{\text{syn}}(\theta) := \mathbb{E}_{x\sim q_{\theta_r,\kappa}}[l_\theta(X)]$,
  $\theta^* \in arg\min_\theta \mathcal{R}_{\text{data}}(\theta)$
- Define

$$\phi_\theta(x) := \nabla_\theta l_\theta(x),$$

$$H_d := \nabla^2 \mathcal{R}_{\text{data}}(\theta^*) = \mathbb{E}_{p_{\text{data}}}[\partial_\theta \phi_\theta(X)]_{\theta=\theta^*},$$

$$r_d := \nabla_\theta \mathcal{R}_{\text{data}}(\theta)|_{\theta_r},$$

$$r_s := \nabla_\theta \mathcal{R}_{\text{syn}}(\theta)|_{\theta_r}$$

- Let $P \succ 0$ be a preconditioner.[1]

[1] P is the optimizer's preconditioner matrix (e.g., $P = I$ for standard SGD, diagonal scaling matrix for Adam.)

Why Neon Works: Anti-Alignment

- **Alignment Scalar**[2]

$$s := \langle r_d, Pr_s \rangle$$

- **Neon improves under anti-alignment.**

  Short synthetic fine-tuning yields $\theta_s = \theta_r - \alpha Pr_s + O(\alpha^2)$, which Neon reverses: $\theta_{\text{Neon}} = \theta_r + w\alpha Pr_s + O(w\alpha^2)$. [3]

  A Taylor expansion of the risk yields

$$\mathcal{R}_{\text{data}}(\theta_{\text{Neon}}) = \mathcal{R}_{\text{data}}(\theta_r) + w\alpha s + \frac{(w\alpha)^2}{2} r_s^\top P^\top \nabla^2 \mathcal{R}_{\text{data}}(\theta_r) Pr_s + O((w\alpha)^3).$$

  When $s < 0$, the negative linear term dominates for small $w > 0$, ensuring that $\mathcal{R}_{\text{data}}(\theta_{\text{Neon}}) < \mathcal{R}_{\text{data}}(\theta_r)$.

[2] See 13

[3] α is a learning rate

Decomposing the Gradients: $s < 0$?

- Let $\epsilon = \theta_r - \theta^*$ be the current model error.
- By taking a Taylor expansion around the true optimum θ^*, we can decompose the real and synthetic gradients:

$$\text{Real Gradient: } r_d \approx H_d \epsilon$$

$$\text{Synthetic Gradient: } r_s \approx H_d \epsilon + b$$

- Sampler Bias

$$b = \mathbb{E}_{q_{\theta_r,\kappa}}[\phi_{\theta^*}(X)]$$

  - If the sampler is perfect ($q = p_{\text{data}}$), $b = 0$.
  - Thus, b is the pure gradient distortion created by the sampler.

- Question:
  - The alignment s is ultimately determined by the relationship between the true signal ($H_d \epsilon$) and the bias (b).
  - *Does bias b help or hinder the correction of error ϵ?*

Theorem 1 : Condition for Anti-Alignment

- To formally answer this, define the angle φ between the model error ϵ and the sampler bias b in the H_d-geometry:

$$\cos\varphi := \frac{\langle \epsilon, H_d^{-1} b \rangle_{H_d}}{\|\epsilon\|_{H_d} \|H_d^{-1} b\|_{H_d}} \in [-1, 1]$$

  - Intuitively, $\cos\varphi < 0$ means that the sampler's bias points in a direction opposing the current error correction.

- Now, let

$$\Delta := \mathbb{E}_{q_{\theta_r,\kappa}}[J_{\theta^*}(X)] - \mathbb{E}_{p_{\text{data}}}[J_{\theta^*}(X)], \;\; J_{\theta^*}(x) := \partial_\theta \phi_\theta(x)|_{\theta^*}$$

$$\eta_0 := \|b\|_{H_d^{-1}}, \quad \eta_1 := \|\Delta\|_{\text{op}, H_d^{-1}}$$

Theorem 1 : Condition for Anti-Alignment

Theorem (Anti-alignment under inference mismatch)

Let $K := H_d^{1/2} P H_d^{1/2}$ *with* $mI \preceq K \preceq MI$.

$$s \leq \underbrace{M(1+\eta_1)\|\epsilon\|_{H_d}^2}_{\text{Positive restoring force}} - \underbrace{m\eta_0\|\epsilon\|_{H_d}[-\cos\varphi]_+}_{\text{Negative pull from bias}} + O(\|\epsilon\|_{H_d}^3)$$

- **Implication**:
  - The first term is the natural restoring force, and the second is the pull from the sampler bias.
  - For a sufficiently small error $\|\epsilon\|_{H_d}$, the linear term dominates.
  - Therefore, **if** $\cos\varphi < 0$**, we are guaranteed that** $s < 0$**.**

Theorem 2 : Mode-Seeking Samplers Induce Anti-alignment

- **Mode-Seeking Samplers?**
  - Typical inference routines that favor high-probability regions.
  - e.g., low temperature, top-k, Classifier-Free Guidance (CFG).

Theorem (Mode-seeking samplers induce $\cos\phi < 0$)

If the sampler q is mode-seeking, then $\cos\phi < 0$ to first order in $\|\epsilon\|_{H_d}$, guaranteeing anti-alignment near good models.

Appendix

Appendix: Alignment scalar

- The Step Vector (Pr_s)
  - During synthetic fine-tuning, the optimizer does not simply move in the direction of r_s. It applies the preconditioner P (e.g., variance normalization in Adam).
  - Thus, $\Delta\theta \propto -Pr_s$ is the **actual step vector** taken in the parameter space.
- The True Gradient (r_d)
  - $r_d = \nabla\mathcal{R}_{\text{data}}(\theta_r)$ evaluates the steepest ascent direction of the true objective landscape.
- The Directional Derivative
  - The inner product computes the first-order change in the true risk when taking a step along Pr_s:

$$\mathcal{R}_{\text{data}}(\theta_r - \alpha Pr_s) \approx \mathcal{R}_{\text{data}}(\theta_r) - \alpha\langle\nabla\mathcal{R}_{\text{data}}(\theta_r), Pr_s\rangle = \mathcal{R}_{\text{data}}(\theta_r) - \alpha s$$

  - P is only applied to r_s because Pr_s is the *footstep* we take, while r_d evaluates the *landscape*.

Proof Sketch of the Theorem 2 : The Covariance Trick

1. Define a Scalar Error $B(x)$

$$B(x) := -\epsilon^\top \phi_{\theta^*}(x) \quad (\text{e.g., } \epsilon^\top \nabla_\theta \log p_\theta(x)|_{\theta=\theta^*})$$

$B(x)$ represents how much the current model θ_r *overestimates* the log-likelihood of sample x due to the error ϵ.

2. Taylor Expansion of Log-likelihood

$$\log p_{\theta_r}(x) \approx \log p_{\theta^*}(x) + B(x)$$

3. Mode-Seeking Weights
   - The synthetic distribution applies a weight to the true distribution: $q(x) \propto w(x) p_{\theta^*}(x)$.
   - For mode-seeking samplers, $w(x)$ is a **monotonically non-decreasing** function of $\log p_{\theta_r}(x)$.
   - Therefore, $w(x)$ is also a **non-decreasing function of** $B(x)$.

Proof Sketch of the Theorem : The Covariance Trick

4. Evaluating the Expected Error under q

$$\mathbb{E}_q[B(X)] \propto \mathbb{E}_{p_{\text{data}}}[w(X)B(X)]$$

Recall that at the true optimum, $\mathbb{E}_{p_{\text{data}}}[B(X)] = -\epsilon^\top \mathbb{E}_{p_{\text{data}}}[\phi_{\theta^*}(X)] = 0$.

Thus, the expectation of the product becomes a Covariance:

$$\mathbb{E}_{p_{\text{data}}}[wB] = \text{Cov}_{p_{\text{data}}}(w(X), B(X))$$

5. Monotone Covariance Inequality
   - Since $w(x)$ and $B(x)$ increase together, their covariance is strictly positive.

   $\Rightarrow$ $\mathbb{E}_q[B(X)] > 0$. (The synthetic samples are biased towards the error).

6. Conclusion

$$\langle \epsilon, b \rangle = \mathbb{E}_q[\epsilon^\top \phi_{\theta^*}(X)] = -\mathbb{E}_q[B(X)] < 0 \implies \cos\varphi < 0$$