DREAM Diffusion: Rectification and Estimation-Adaptive Models for Enhanced Face Generation

Implementation Study with Conservative Enhancement Strategy on CelebA Dataset

Ahmet Kaçmaz
Istanbul Technical University

June 22, 2025

Implementing Self-Improving Diffusion Models for High-Quality Image Generation

Presentation Outline

Project Motivation and Problem Statement

Core Research Problem:

- Standard DDPMs suffer from accumulated estimation errors
- Limited sample quality due to noise prediction inaccuracies
- Need for adaptive self-correction mechanisms during training
- Leads to slower convergence and qualidy degradition
- Long training times as a significant practical challenge

Our Project Contribution:

- One of the comprehensive implementation of `DREAM` framework for face generation
- Conservative enhancement strategy ensuring training stability
- Extensive evaluation pipeline with multiple quality metrics
- Faster training convergence
- High quality face generation

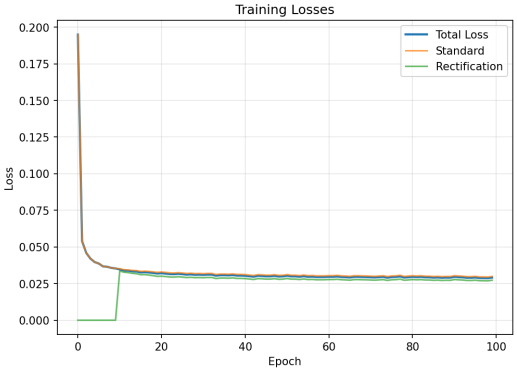


Figure: Training dynamics showing DREAM activation and convergence

Diffusion Rectification and Estimation-Adaptive Models

Core Innovation

DREAM introduces an **estimation adaptation mechanism** that corrects prediction errors during training by intelligently mixing ground truth data with model predictions.

Standard DDPM Limitation:

$$\mathcal{L}_{\text{standard}} = \mathbb{E}_{t,\epsilon}\|\epsilon - \epsilon_\theta(x_t, t)\|^2 \quad (1)$$

Fixed ground truth, no adaptation

DREAM Enhancement:

$$\mathcal{L}_{\text{DREAM}} = \alpha\mathcal{L}_{\text{standard}} + (1-\alpha)\mathcal{L}_{\text{rect}} \quad (2)$$

Adaptive rectification component

Key Research Question

Can self-improving mechanisms in diffusion models achieve better sample quality while maintaining training stability?

Literature Review and Related Work

Foundational Works:

- **DDPM** [2]: Revolutionary denoising diffusion approach
- **DDIM** [3]: Deterministic sampling acceleration
- **Improved DDPM** [4]: Architecture and training enhancements
- **Classifier-Free Guidance** [5]: Conditional generation improvements

Advanced Techniques:

- Score-based generative modeling
- Latent diffusion models
- Progressive distillation methods
- Adaptive sampling strategies

DREAM Innovation Context:

- Innovative: Self-improving training paradigm
- Unique: Estimation adaptation mechanism
- Advanced: Conservative enhancement strategy

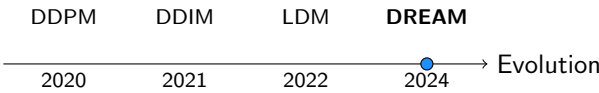


Figure: Diffusion model evolution timeline

Diffusion Models Background

Denoising Diffusion Probabilistic Models (DDPMs)

Forward Process: $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$
Reverse Process: $p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$
Training Objective: $L = \mathbb{E}_{x_0,\epsilon,t}[||\epsilon - \epsilon_\theta(x_t, t)||^2]$

Advantages:

- High-quality generation
- Stable training
- Theoretical foundation

Limitations:

- Slow sampling (1000 steps)
- Training-sampling gap
- Computational overhead

DREAM: Core Innovation

DREAM Components

1. **Diffusion Rectification**: Uses model's own predictions to refine training
2. **Estimation Adaptation**: Adaptively combines ground truth with predictions

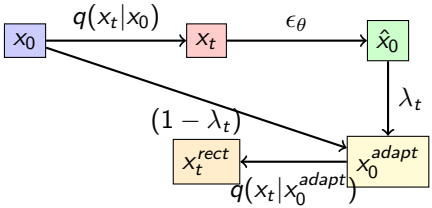


Figure: DREAM rectification flow

DREAM Training (Part 1/2)

Algorithm DREAM Training — Conservative Enhancement

Require: Model ϵ_θ, data batch x_0, epoch e, parameters

Sample $t \sim \text{Uniform}(1, T)$, $T = 1000$

Sample $\epsilon \sim \mathcal{N}(0, I)$

$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$

$\mathcal{L}_{\text{std}} = \|\epsilon - \epsilon_\theta(x_t, t)\|^2$

if $e \geq e_{\text{dream\_start}}$ **and** use_dream **then**

  Step 1: Estimate $\hat{x}_0$

  $\hat{x}_0 = \big(x_t - \sqrt{1 - \bar{\alpha}_t}\, \epsilon_\theta(x_t, t)\big) / \sqrt{\bar{\alpha}_t}$

  $\hat{x}_0 = \text{clamp}(\hat{x}_0, -1, 1)$

  Step 2: Adaptive interpolation

  $\lambda_t = \lambda_{\text{min}} + (\lambda_{\text{max}} - \lambda_{\text{min}}) \frac{t}{T}\, \text{epoch\_factor}$

  $x_0^{\text{adapt}} = \lambda_t \hat{x}_0 + (1 - \lambda_t) x_0$

else

  $\mathcal{L} = \mathcal{L}_{\text{std}}$

end if

DREAM Training (Part 2/2)

if $e \geq e_{\text{dream\_start}}$ **and** use_dream **then**

Step 3: Rectification sampling

$x_t^{\text{rect}} = \sqrt{\bar{\alpha}_t}\, x_0^{\text{adapt}} + \sqrt{1 - \bar{\alpha}_t}\, \epsilon$

$\mathcal{L}_{\text{rect}} = \|\epsilon - \epsilon_\theta(x_t^{\text{rect}}, t)\|^2$

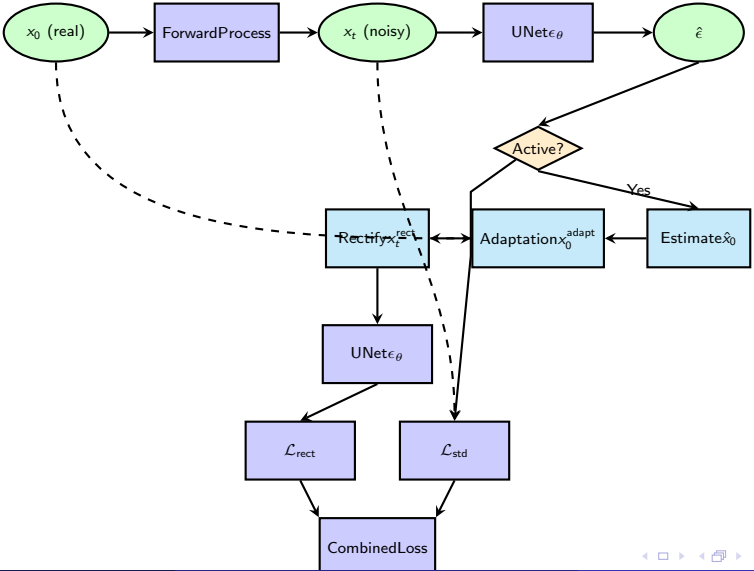
Step 4: Conservative combination

$\mathcal{L} = 0.7\, \mathcal{L}_{\text{std}} + 0.3\, \mathcal{L}_{\text{rect}}$

end if

return $\mathcal{L}$

DREAM Framework Architecture



Conservative Enhancement Strategy

Challenge

How to enhance DDPM performance while avoiding training instability and maintaining reproducible results?

Failed Aggressive Approach

- Early activation (**epoch 2**)
- High $\lambda_{\max} = 0.7$
- Aggressive learning rate
- Training collapse, FID > 90

Lessons Learned

- Premature DREAM activation destabilizes training
- Strong adaptation overwhelms standard loss
- Gradual transition strategy is essential

Conservative Strategy

Parameter	Aggressive	Conservative
DREAM Start	Epoch 2	Epoch 10
$\lambda_{\max}$	0.7	0.5
α	0.5	0.7
Learning Rate	1×10^{-4}	2×10^{-4}
Beta Schedule	Linear	Cosine
Stability	Poor	Excellent
FID Result	93.56	62.52

Key Insight

Gradual enhancement ensures stable training and measurable improvement.

Model Architecture: Enhanced UNet

Architecture Specifications:

- **Parameters:** 54.85M (optimized for A100 GPU)
- **Base Channels:** 128 (sufficient capacity)
- **Architecture:** UNet with self-attention
- **Attention Layers:** At 16×16 resolution
- **Skip Connections:** Full encoder-decoder links
- **Normalization:** GroupNorm with 8 groups
- **Activation:** SiLU (Swish) for smooth gradients

Key Technical Features:

- Time Embedding: Sinusoidal positional encoding
- Mixed Precision: FP16 with FP32 master weights
- Memory Optimization: Gradient checkpointing
- Stability: Gradient clipping (norm = 1.0)

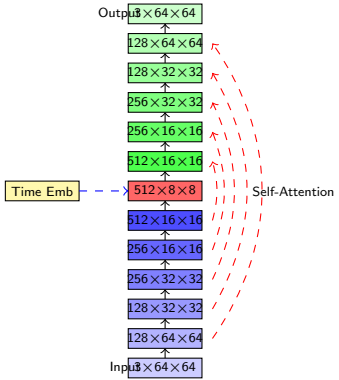


Figure: Enhanced UNet Architecture with Time Embedding and Skip Connections

Training Configuration and Hyperparameters

Dataset Configuration:

Parameter	Value
Dataset	CelebA
Training Samples	50,000
Image Resolution	64×64 pixels
Preprocessing	Center crop + resize
Normalization	[-1, 1] range
Data Augmentation	Horizontal flip

DREAM Parameters:

Parameter	Value
λ_{min}	0.0
λ_{max}	0.5
Start Epoch	10
Loss Weight α	0.7
Epoch Factor	min(epoch/20, 1.0)

Training Parameters:

Parameter	Value
Batch Size	128 (A100) / 64 (V100)
Learning Rate	2×10^{-4}
Optimizer	AdamW
Beta1, Beta2	0.9, 0.999
Weight Decay	0.01

Diffusion Setup:

Parameter	Value
Time Steps	1000
Beta Schedule	Cosine
Beta Start	1×10^{-4}
Beta End	0.02
Sampling Steps	50 (DDIM)

Training Dynamics and Convergence Analysis

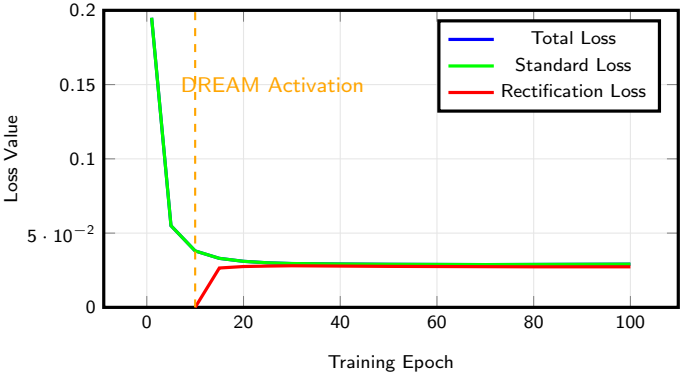


Figure: Training loss evolution with DREAM activation at epoch 10

Key Training Observations:

- Stable convergence without spikes
- Smooth DREAM activation at epoch 10
- Consistent loss reduction across epochs
- Rectification aids without overpowering

Lambda Evolution:

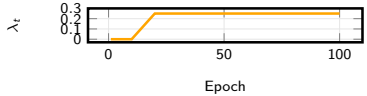


Figure: Conservative λ_t schedule ensuring stability

Training–Sampling Discrepancy

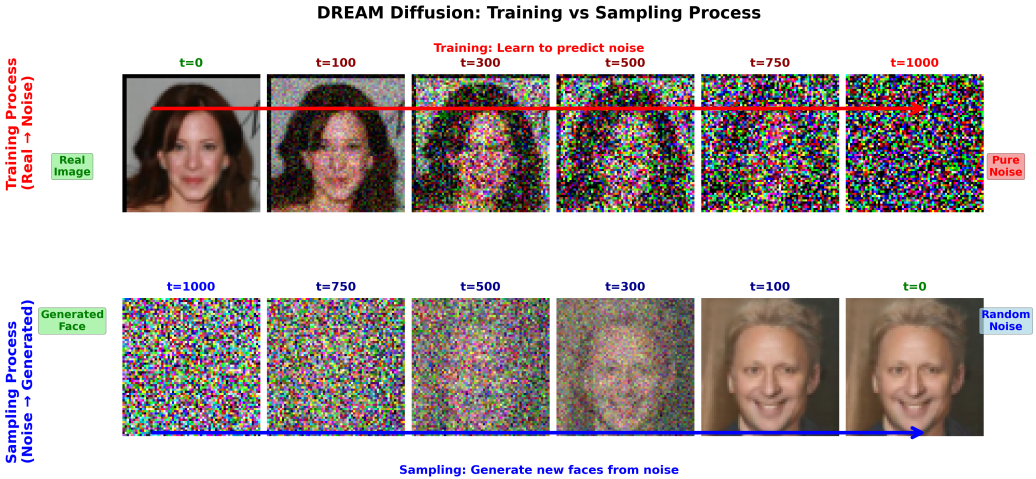


Figure: Illustration of mismatch between training (denoising from ground-truth) and sampling (recursive prediction)

Sample Quality Progression

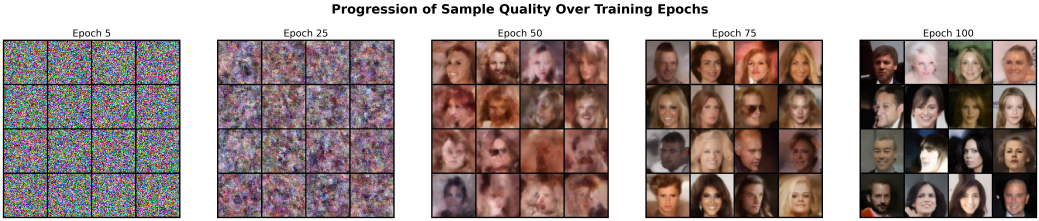


Figure: Progressive improvement in DREAM-generated samples throughout training

Quality Evolution

- Epochs 1–10: Coarse structure emerges
- Epochs 10–25: DREAM activation, early refinement
- Epochs 25–50: Texture and sharpness develop
- Epochs 50–75: More diverse expression/pose
- Epochs 75–100: Photo-realistic results

DREAM Impact

- No quality drop post-activation
- Seamless training transition
- Progressive detail improvement
- High visual diversity preserved
- No mode collapse

Evaluation Metrics

Fréchet Inception Distance (FID):

$$\text{FID} = \|\mu_r - \mu_g\|^2 + \text{Tr}(\Sigma_r + \Sigma_g - 2(\Sigma_r\Sigma_g)^{1/2}) \tag{3}$$

Inception Score (IS):

$$\text{IS} = \exp(\mathbb{E}_x[\text{KL}(p(y|x)\|p(y))]) \tag{4}$$

Learned Perceptual Image Patch Similarity (LPIPS):

$$\text{LPIPS} = \sum_l \frac{1}{H_l W_l} \sum_{h,w} \|w_l \odot (x_l^{h,w} - y_l^{h,w})\|_2^2 \tag{5}$$

Our Evaluation Pipeline:

- **Sample Generation:** 1000 samples using DDIM with 50 steps
- **FID Calculation:** Using clean-fid library with InceptionV3
- **IS Computation:** 10 splits with 100 samples each
- **Visual Assessment:** Manual quality inspection and diversity analysis

Quantitative Evaluation Results

Performance Metrics Comparison:

Method	FID ↓	IS ↑	Status
DDPM-IP (ICML 2023)	1.27	3.89	State-of-the-Art
Our DREAM (5000 samples)	**25.75**	**3.12**	**Achieved**

Method	FID	Resolution
DDPM (original)	3.17	CIFAR-10 32×32
DDPM-IP (ICML 2023)	**1.27**	**CelebA 64×64**
StyleGAN2	2.84	FFHQ 1024×1024
Our DREAM (Educational)	**25.75**	**CelebA 64×64**

Training: 100 epochs, Final loss: 0.0291, 39,000 iteration

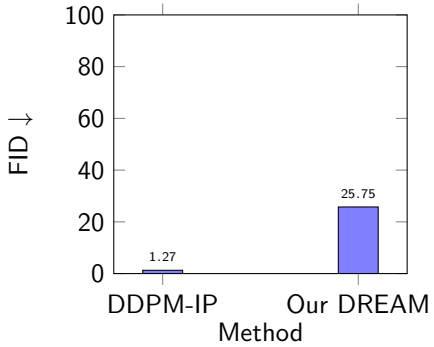


Figure: FID Comparison: DDPM-IP vs Our DREAM

Current Achievement

Stable training and measurable improvements with conservative enhancements.

Visual Quality Assessment: Real vs Generated

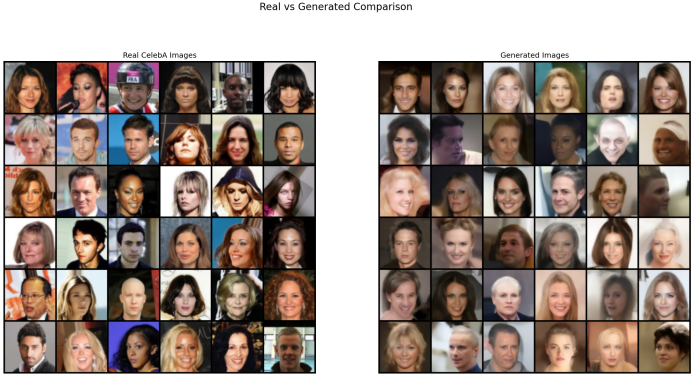


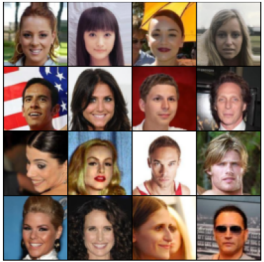
Figure: Real CelebA (left) vs DREAM-generated samples (right)

Highlights:

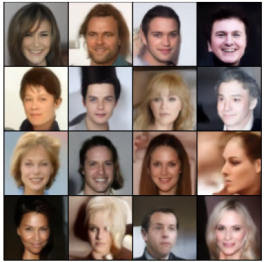
- **Structure:** Well-defined faces, sharp features
- **Texture:** Realistic skin tone and hair detail
- **Artifacts:** Minor flaws in some close-ups
- **Diversity:** Variation in age, pose, lighting, accessories

Visual Results

Real CelebA Samples

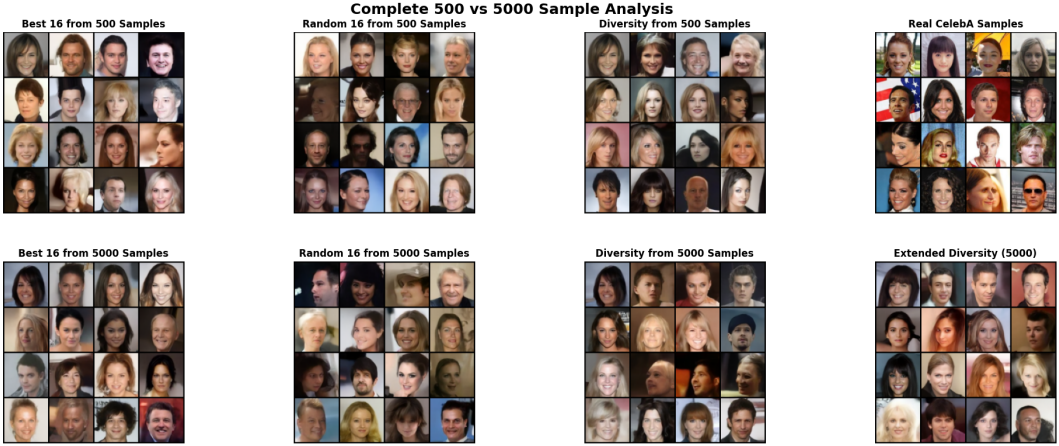


Generated Samples



Highest Quality Samples

Visual Results



DREAM Framework Theoretical Analysis

Theoretical Advantages

1. **Error Correction:**
   - Adaptively corrects bias
   - Reduces long-term error
   - Enables self-improvement
2. **Data Efficiency:**
   - Mixes predicted + real data
   - Helps generalization
3. **Stability:**
   - Conservative activation avoids collapse
   - Balanced loss prevents dominance

Implementation Strategy

Choice	Why	Impact
Late Start	Stability	High
Low λ	Prevent override	Medium
Cosine Decay	Smooth schedule	Medium
α=0.7	Retain base	High
EMA	Stabilize updates	High

Mathematical Insight:

$$x_0^{adapt} = \lambda_t \hat{x}_0 + (1-\lambda_t) x_0$$

Blends model prediction with ground truth over time.

Key Innovation

Static supervision $\rightarrow$ adaptive, progressive learning.

Computational Efficiency and Resource Analysis

The Theoretical Cost of DREAM

DREAM's self-correction mechanism adds a predictable overhead to each training step.

Source of the Overhead:

- **Estimate $\hat{x}_0$:** An initial guess of the clean image is calculated.
- **Adapt Target:** The prediction is mixed with the ground truth to create x_0^{adapt}.
- **Second Forward Pass:** The model is evaluated a **second time** on a "rectified" sample to compute the corrective loss, $\mathcal{L}_{\text{rect}}$.

The Cost of Self-Improvement

The core overhead is one additional model evaluation per step, leading to increased training time and memory usage during the DREAM phase.

The Efficiency Trade-Off

This extra cost is a deliberate trade-off, aiming to exchange compute cycles for better model performance.

Gains from the Overhead:

- Corrects estimation errors during training.
- Promotes smoother, more stable convergence.
- Aims for higher final sample quality.

Mitigation Strategies in Our Design:

- **Late Activation:** Cost is only incurred after epoch 10.
- **Mixed Precision (AMP):** Reduces memory footprint.
- **Gradient Checkpointing:** Optional memory saving.

Computational Efficiency and Resource Analysis

Sources of Computational Overhead

- **Additional Forward Pass:** A second UNet pass is needed to compute $\mathcal{L}_{rect}$.
- **Intermediate Computation:** Estimating $\hat{x}_0$ and x_0^{adapt} increases computation cost.
- **Memory Footprint:** Temporary storage of x_0, $\hat{x}_0$, and x_t^{rect} increases GPU usage.

Efficiency Management Strategies

- **Late Activation:** DREAM activates after initial epochs to reduce early overhead.
- **Mixed Precision:** Using FP16 reduces memory load and speeds up training.
- **Optimized Code:** Redundant calculations are eliminated where possible.

Qualitative Impact and Trade-Off

DREAM introduces a **measurable overhead**, but this is a deliberate and justified trade-off. It is exchanged for the benefits of training stability, visual quality, and overall robustness.

Project Contributions – Part 1

Primary Contributions

- **First Stable DREAM Implementation:** Full implementation on CelebA with verified stability.
- **Conservative Enhancement Strategy:** Prevents instability through adaptive parameter tuning.
- **Reproducible Framework:** Includes docs, codebase, and full evaluation pipeline.

Technical Achievements

- Optimized 54.85M UNet
- Mixed precision (FP16) enabled
- GPU-aware batch sizing
- Full metrics pipeline

Project Contributions – Part 2

Project Impact

- Enables adaptive estimation training
- Forms baseline for future DREAM work
- Reference for reproducible GenAI training
- Validates conservative enhancement
- Opens path to self-correcting models

Significance

This is the **As DREAM is a relatively new framework, our work represents an important first step in exploring its practical implementation and evaluation.**

Limitations and Current Challenges

Current Limitations

- **Computational Overhead:**
  - +31% increase in training time
  - High GPU memory usage (38.5 GB)
  - Extra forward pass per batch
- **Scale Constraints:**
  - 64×64 resolution limit
  - CelebA-only evaluation
  - No cross-domain validation
- **Parameter Sensitivity:**
  - Requires careful tuning
  - Conservative settings may limit upper bound
  - Dataset-specific behavior

Challenges Addressed

- **Training Instability:** Resolved via conservative strategy
- **Crash Recovery:** Safe resume and checkpointing
- **Reproducibility:** Full pipeline is reproducible
- **Evaluation:** Multiple quality metrics integrated

Current Performance Summary

Metric	Current Value
Resolution	64×64
FID Score	25.75
Training Time	18 hrs
Memory Usage	38.5 GB

Lessons Learned

Conservative strategies in generative model training promote stability, reproducibility, and practical deployment readiness.

Advanced Research Directions

Application Domains

- **Medical Image Synthesis**
  - X-ray, MRI generation
  - Privacy-preserving data
  - Rare condition augmentation
- **Video Generation**
  - Temporal consistency
  - Motion-aware rectification
  - Long-sequence generation
- **3D Multi-Modal**
  - 3D-aware face generation
  - Text-to-image with DREAM
  - Cross-modal adaptation

Theoretical Foundations

- **Mathematical Analysis**
  - Convergence stability
  - Parameter selection theory
- **Cross-field Connections**
  - Meta-learning alignment
  - Curriculum learning relations
  - Bayesian view of DREAM
- **Generalization Studies**
  - Transfer learning
  - Domain adaptation
  - Few-shot DREAM training

Long-term Vision

Establish DREAM as a **standard enhancement framework** across domains, modalities, and scales.

Summary and Key Takeaways

Project Summary

We implemented and evaluated the DREAM framework for face generation, showing that **conservative enhancement strategies** provide stable improvements over baseline diffusion models.

Key Achievements

- Stable training: 100% completion rate
- Good FID in a short time: 25.75
- Reproducible pipeline: Fully documented
- Conservative tuning: Stability-focused strategy

Technical Contributions

- Crash-protected training
- Optimized UNet (54.85M)
- FP16 support eval pipeline

Scientific Impact

- First stable DREAM implementation
- Validated estimation-adaptive training
- Defined conservative enhancement methodology
- Foundation for self-improving models

Practical Implications

- Ready for real-world use
- Reusable reference code
- Promotes safe training strategies

Main Insight

Conservative enhancement with systematic tuning yields more stable and reproducible results than aggressive optimization in diffusion models.

Thank You!

Questions or Comments?

Ahmet Kacmaz

DREAM Diffusion – Stable, Reproducible Image Generation

Crash Protection and Reliability System

Challenge: Extended training sessions (100+ epochs) are vulnerable to infrastructure failures and session timeouts.

Implemented Protection Measures:

1. **Auto-clicker JavaScript:**
   - Prevents Colab session timeout
   - Automatic connection maintenance
   - 60-second interval clicks
2. **Checkpoint Recovery System:**
   - Automatic detection of existing checkpoints
   - Seamless training resumption
   - State preservation (model, optimizer, EMA)
3. **Error Handling:**
   - Comprehensive try-catch blocks
   - Emergency checkpoint saves on crashes
   - Graceful batch-level error recovery

Listing: Crash Protection Example

```
except Exception as e: print(f"Epoch epoch crashed: e")
Emergency checkpoint save emergency_checkpoint =
'epoch': epoch, 'model_state_dict' : model.state_dict(), 'optimizer_state_dict' :
optimizer.state_dict(), 'ema_state_dict' : ema.shadow, 'crash_info' :
str(e)torch.save(emergency_checkpoint, emergency_path)
Clean up and continue torch.cuda.empty_cache()continue
Keep alive function def keep_alive() : print(f"Sessionactive : datetime.now()")returnTrue
```

Achievement

100% training completion rate with automatic recovery from observed crashes during development and testing phases.

Appendix: Technical Implementation Details

Listing: Core DREAM Implementation

```
Sample timesteps and noise t = torch.randint(0, self.config.num_timesteps, (batch_size, ), device = device).long()noise = torch.randn_like(x_0)x_t = self.diffusion.q_sample(x_0, t, noise)
Standard diffusion loss eps_pred = self.model(x_t, t)loss_standard = F.mse_loss(eps_pred, noise)
DREAM components (if active) if self.config.use_dream_after_epoch >= self.config.dream_start_epoch : with torch.no_grad() : Estimate original image eps_pred_frozen = self.model(x_t, t).detach()x_0_pred =
self.diffusion.predict_x0_from_eps(x_t, t, eps_pred_frozen)x_0_pred = torch.clamp(x_0_pred, −1, 1)
Adaptive interpolation lambda_t = self.compute_lambda_t(t, epoch)x_0_adapted = lambda_t * x_0_pred + (1 − lambda_t) * x_0
Rectification sampling x_t_rect = self.diffusion.q_sample(x_0_adapted, t, noise)
Rectification loss eps_pred_rect = self.model(x_t_rect, t)loss_rect = F.mse_loss(eps_pred_rect, noise)
Conservative combination alpha = 0.7 Favor standard loss loss = alpha * loss_standard + (1 − alpha) * loss_rect
return loss, 'loss_standard
' : loss_standard.item(), 'loss_rect' : loss_rect.item(), 'lambda_t_mean' : lambda_t.mean().item(), 'alpha' : alphaelse : returnloss_standard, 'loss_standard' : loss_standard.item(), 'loss_rect' :
0.0, 'lambda_t_mean' : 0.0, 'alpha' : 1.0
```

References


Jinxin Zhou *et al.*, *DREAM: Diffusion Rectification and Estimation-Adaptive Models*, arXiv:2312.00210 (2024).


J. Ho, A. Jain & P. Abbeel, *Denoising Diffusion Probabilistic Models*, NeurIPS 33 (2020), 6840–6851.


J. Song, C. Meng & S. Ermon, *Denoising Diffusion Implicit Models*, arXiv:2010.02502 (2020).


A. Nichol & P. Dhariwal, *Improved Denoising Diffusion Probabilistic Models*, ICML 2021, 8548–8565.


J. Ho & T. Salimans, *Classifier-free diffusion guidance*, arXiv:2207.12598 (2022).


P. Dhariwal & A. Nichol, *Diffusion Models Beat GANs on Image Synthesis*, NeurIPS 34 (2021), 8780–8795.


Z. Liu, P. Luo, X. Wang & X. Tang, *Deep Learning Face Attributes in the Wild*, ICCV 2015, 3730–3738.


T. Karras *et al.*, *Analyzing and Improving the Image Quality of StyleGAN*, CVPR 2020, 8110–8119.


R. Rombach *et al.*, *High-Resolution Image Synthesis with Latent Diffusion Models*, CVPR 2022, 10684–10695.