Memory-Conditioned Diffusion Model for Tail-Class Feature Hallucination

# 1. Overview

This document outlines the design of a memory-conditioned diffusion model built on top of the Cost-Sensitive Loss (CSL) framework. The model synthesizes features for tail classes by training a class-conditional diffusion model using stored feature representations. These synthetic features are used to augment the training set, and the CSL loss is adapted to weight their contribution based on confidence measures.

# 2. Core Components

## 2.1 Feature Memory Bank

For each class c, maintain a rolling memory of feature vectors (f\_i = φ(x\_i)). The memory is updated using exponential moving average or reservoir sampling.

Update rule: M\_c ← (1 - α) \* M\_c + α \* φ(x\_i)

## 2.2 Diffusion Model (DDPM for Features)

### a. Forward Process (Noise Addition)

Add Gaussian noise across T steps to clean feature vectors.  
q(f\_t | f\_{t-1}) = N(f\_t; sqrt(1 - β\_t) \* f\_{t-1}, β\_t \* I)  
Full marginal: q(f\_t | f\_0) = N(f\_t; sqrt(ᾱ\_t) \* f\_0, (1 - ᾱ\_t) \* I)

### b. Reverse Process (Denoising)

Train ε\_θ(f\_t, t, c) to predict noise at step t for class c.  
Loss: L\_denoise = E[||ε - ε\_θ(f\_t, t, c)||²]

### c. Architecture of ε\_θ

Input: f\_t ∈ ℝ^d, timestep t, class embedding e\_c  
Model: MLP (Linear → ReLU → Linear), optional Transformer block  
Use positional embeddings and concatenate class embeddings

## 2.3 Sampling Synthetic Features

Start from Gaussian noise and iteratively denoise using reverse process:  
f\_{t-1} = 1/sqrt(1 - β\_t) \* (f\_t - β\_t/sqrt(1 - ᾱ\_t) \* ε\_θ(f\_t, t, c)) + σ\_t \* z

## 2.4 Confidence-Adaptive CSL

Measure similarity between generated and real features to compute confidence:  
conf(𝑓̃) = cos(𝑓̃, μ\_c) or entropy(softmax(W𝑓̃))  
γ\_c^adaptive = γ\_c^CSL \* conf(𝑓̃)  
Use synthetic features to augment minibatches for tail classes

# 3. Evaluation of Feature Quality

Quantitative:  
- Accuracy with/without generated features  
- Diversity metrics (mean distance, spread)  
- Class-wise F1-score  
- KL divergence real vs. synthetic

Qualitative:  
- t-SNE visualization  
- (Optional) Visual samples if extended to image space

# 4. Optional Extensions

- Use Transformer instead of MLP for ε\_θ  
- Replace MSE with InfoNCE loss  
- Extend to image-level diffusion using semantic prompts

A diagram of a model of a bird

AI-generated content may be incorrect.

image-level diffusion using semantic prompts

**Motivation**

* Feature-level augmentation is effective but **lacks visual diversity**.
* Image-level synthesis can **complement** feature-level CSL by:
  + Enriching training with visually plausible samples.
  + Bridging representation learning with generative modeling.
  + Allowing interpretability through visualization of tail-class diversity.

**🎯 Goal**

Use **semantic prompts derived from stored CSL features** to guide a **diffusion-based image generator** (e.g., Stable Diffusion, Imagen, or CustomDiffusion) to hallucinate images that:

* Belong to tail classes.
* Reflect the semantic characteristics learned from head/tail feature representations.
* Are used directly or indirectly (via re-embedding) to augment training.

**🔧 Methodological Design**

**1. Semantic Prompt Extraction**

From CSL training, you have a memory bank Mc\mathcal{M}\_cMc​ per class.

**a. Option 1: Use Class Text Labels Directly**

* Prompt = "a photo of a {class\_name}"
* Standard in CLIP/BLIP-based pipelines.

**b. Option 2: Use Feature-to-Text Embedding Projection**

* Use a **feature-to-text decoder** (e.g., BLIP or Transformer mapper) to generate textual prompts from CSL features:

promptc=BLIP−1(μc)\text{prompt}\_c = \text{BLIP}^{-1}(\mu\_c)promptc​=BLIP−1(μc​)

**c. Option 3: Use Nearest Visual Exemplars + Caption**

* Find nearest images to μc\mu\_cμc​ in CLIP space.
* Use BLIP to caption them: "A bird with green wings and red beak"

**2. Image-Level Diffusion with Class Conditioning**

Use a model like **Stable Diffusion v2**, **Imagen**, or **CustomDiffusion**:

* Fine-tune or guide the model to generate images based on semantic prompts for **tail classes** only.
* Use **LoRA**, **DreamBooth**, or **CustomDiffusion** to personalize the generation toward long-tail domains.

**3. Training Integration Approaches**

**a. Pseudo-Labeling + Fine-Tuning**

* Generate images for tail classes.
* Use a pretrained backbone to embed these images.
* Add synthetic (image, label) pairs to the training pool.

**b. Joint CSL + Image Augmentation**

* CSL training loop samples a mix of:
  + Real images
  + Synthetic images
  + Synthetic features (from memory-conditioned feature-level diffusion)

**c. Curriculum Learning**

* Initially train on real head+tail data.
* Inject synthetic tail images later to prevent overfitting or bias.

**📊 Evaluation Metrics**

* **FID/KID** on generated tail-class images vs. real ones
* **Top-1 Accuracy on tail classes** with/without image-level augmentation
* **Diversity metrics**: intra-class perceptual distance
* **CLIP similarity** between prompt and generated image

**🧩 Challenges & Solutions**

| **Challenge** | **Solution** |
| --- | --- |
| Generating consistent tail-class images | Use prompt tuning, LoRA, or class embedding |
| Semantic drift in diffusion | Condition generation on CSL-derived features |
| Evaluation of quality | Use CLIP similarity + classifier performance boost |

**🏁 Final Remarks**

This extension enables:

* End-to-end **data synthesis pipeline** for low-resource classes.
* Training with **multi-modal augmentation** (features + images).
* Insightful **visual diagnostics** of generative coverage across tail classes.

Implementation Details:

### Phase 1: Memory Bank Construction

**Input:** Training data ( {(x\_i, y\_i)}\_{i=1}^N ), pretrained encoder ( () ), class set ( )

**For each input sample ( (x\_i, y\_i = c) ):** 1. Extract feature vector:  
( f\_i = (x\_i) ^d ) 2. Update memory bank ( \_c ):  
If using **Exponential Moving Average (EMA)**:  
[ \_c (1 - ) \_c + f\_i ] If using **Reservoir Sampling**, maintain a bounded memory set of representative ( f\_i ) vectors per class ( c )

### Phase 2: Training the Memory-Conditioned Diffusion Model

**Objective:** Learn to denoise feature vectors ( f \_c ) via a DDPM conditioned on class identity.

#### 2.1. Forward Diffusion Process

For a feature ( f\_0 \_c ), define: - Time-step ( t ) - Noise schedule ( \_t ) - Cumulative product ( {}*t =* {s=1}^t (1 - \_s) )

Sample: [ f\_t = f\_0 + , (0, I) ]

#### 2.2. Reverse Process Learning

Train a neural network ( \_(f\_t, t, c) ) to predict ( )

**Loss function:** [ *{} =* {f\_0, t, } ]

**Model Architecture:** - Inputs: ( f\_t ), timestep ( t ), class embedding ( e\_c ) - Architecture: MLP or Transformer, optionally using positional encodings for ( t )

### Phase 3: Generating Synthetic Features

**For a target tail class ( c ):**

1. Sample ( f\_T (0, I) )
2. Iteratively compute reverse steps from ( T ) to 1: [ f\_{t-1} = ( f\_t - \_(f\_t, t, c) ) + \_t z\_t ] where ( z\_t (0, I) )
3. Final result ( f\_0 ) is the synthetic feature vector for class ( c )

### Phase 4: Confidence-Adaptive CSL Training

**Objective:** Adjust the influence of synthetic features based on confidence.

#### 4.1. Compute Confidence for Synthetic Feature ( ):

* **Option 1 (Cosine similarity):**  
  [ () = (, \_c), \_c = \_c ]
* **Option 2 (Entropy):**  
  Compute softmax scores ( p = (W ) ),  
  then ( () = -p\_i p\_i )

#### 4.2. Update Class Weight in CSL:

[ \_c^{} = \_c^{} () ]

#### 4.3. Use in CSL Training Loop:

* For each minibatch:
  + Sample real features from head + tail classes
  + Add synthetic features for tail classes, weighted by ( \_c^{} )

### Phase 5: Optional Image-Level Extension via Diffusion

#### 5.1. Semantic Prompt Extraction for Class ( c ):

* Option 1: Use class label ( ) “a photo of a {class\_name}”
* Option 2: Use inverse mapping ( ^{-1}(\_c) )
* Option 3: Find nearest image exemplars in CLIP space and caption them

#### 5.2. Image Synthesis Using Prompt-Guided Diffusion

* Use Stable Diffusion / Imagen / CustomDiffusion
* Inject class conditioning via prompt, LoRA, or DreamBooth
* Generate images for tail class ( c )

#### 5.3. Training Integration Options:

* **Pseudo-labeling:** Embed generated images, treat as labeled data
* **Curriculum augmentation:** Inject synthetic tail images/features after warm-up
* **Joint CSL loop:** Combine real + synthetic images + features

### Phase 6: Evaluation

#### 6.1. Feature Evaluation

* Accuracy on tail classes with/without augmentation
* Diversity: intra-class feature distance
* KL divergence between real vs. synthetic feature distributions
* Visualization: t-SNE plots

#### 6.2. Image Evaluation

* FID / KID between real and synthetic tail-class images
* CLIP similarity between generated image and prompt

Boost in classifier performance

Related Work:

**1. Class-Balancing Diffusion Models (CBDM)**

* **Paper:** *Class-Balancing Diffusion Models* (Qin et al., CVPR 2023) [aaai.org+15openaccess.thecvf.com+15researchgate.net+15](https://openaccess.thecvf.com/content/CVPR2023/papers/Qin_Class-Balancing_Diffusion_Models_CVPR_2023_paper.pdf?utm_source=chatgpt.com)
* **Key idea:** Introduces a distribution-adjustment regularizer to mitigate diversity and fidelity drop for tail classes in diffusion models.
* **Baseline comparisons:**
  + Standard diffusion model (imbalanced training),
  + Oversampling methods,
  + Conditional (guided) generation.
* **Datasets/eval:** CIFAR-100LT, measuring downstream classification benefits and generation quality metrics (e.g. FID, Inception Score).

**2. CORAL: Disentangling Latent Representations in Long-Tailed Diffusion**

* **Paper:** *CORAL* (Rodriguez et al., June 2025) [arxiv.org+1arxiv.org+1](https://arxiv.org/abs/2506.15933?utm_source=chatgpt.com)[arxiv.org](https://arxiv.org/abs/2305.00562?utm_source=chatgpt.com)
* **Key idea:** Uses supervised contrastive latent alignment to encourage separation of tail-class latent subspaces, reducing feature overlap between head and tail.
* **Baseline comparisons:**
  + Standard long-tailed diffusion,
  + CBDM,
  + Possibly other class-conditioning methods.
* **Datasets/eval:** Long-tailed data (e.g. imbalanced CIFAR), evaluating sample quality/diversity on tail classes and representation overlap.

**3. Synthetic Augmentation for Long-Tailed Food Classification**

* **Paper:** *Synthetic Data Augmentation using Pre‑trained Diffusion Models for Long‑tailed Food Image Classification* (Koh et al., June 2025) [github.com+12arxiv.org+12arxiv.org+12](https://arxiv.org/abs/2506.01368?utm_source=chatgpt.com)[arxiv.org](https://arxiv.org/abs/2102.12867?utm_source=chatgpt.com)[arxiv.org](https://arxiv.org/abs/2506.15933?utm_source=chatgpt.com)
* **Key idea:** Generates class-specific synthetic images via prompt-engineered diffusion with positive/negative prompt strategies to boost tail-class diversity and separation.
* **Baseline comparisons:**
  + Fine-tuning diffusion on balanced data,
  + Naïve prompt-based augmentation,
  + Traditional oversampling.
* **Datasets/eval:** Food image datasets, reporting top-1 classification accuracy and diversity metrics post-augmentation.

**4. SeedSelect: Generating Rare Concepts without Fine-Tuning**

* **Paper:** *Generating images of rare concepts using pre‑trained diffusion models* (April 2023) [arxiv.org+1github.com+1](https://arxiv.org/abs/2506.01368?utm_source=chatgpt.com)[arxiv.org+1github.com+1](https://arxiv.org/html/2304.14530v3?utm_source=chatgpt.com)
* **Key idea:** Uses a small set of reference images to steer diffusion sampling toward rare classes (e.g., “pay phone”), improving few-shot generation quality.
* **Baseline comparisons:**
  + Vanilla prompt-based generation,
  + Finetuned models,
  + Other reference-based prompt techniques.
* **Datasets/eval:** Measures faithfulness, diversity, semantic fidelity (e.g., CLIP scores), and impact on downstream few-shot recognition.

**Evaluation Baselines & Metrics**

When assessing a memory-conditioned diffusion model for tail-class feature hallucination, you should consider the following **standard baselines** and **metrics**:

**Baseline Methods**

| **Category** | **Baseline Method** | **Description** |
| --- | --- | --- |
| **Diffusion-based** | Standard Diffusion (imbalanced) | Train DDPM on imbalanced data without balancing. |
|  | CBDM | Adds distribution-adjustment regularizer [arxiv.org+1openreview.net+1](https://arxiv.org/html/2304.14530v3?utm_source=chatgpt.com)[openreview.net+1github.com+1](https://openreview.net/forum?id=dd0rUW29tQ&noteId=7w7VHbJkcn&utm_source=chatgpt.com)[arxiv.org+3arxiv.org+3arxiv.org+3](https://arxiv.org/abs/2305.00562?utm_source=chatgpt.com)[arxiv.org+6arxiv.org+6github.com+6](https://arxiv.org/abs/2506.15933?utm_source=chatgpt.com). |
|  | CORAL | Adds contrastive latent alignment . |
| **Prompt-guided synthesis** | SeedSelect | Uses reference prompts to generate rare classes . |
|  | Synthetic diffusion with prompt negatives | As in food classification paper . |
| **Non-diffusion augmentation** | Feature hallucination (e.g. FASA, Lazarou) | Simple augmentation based on statistics . |
| **Sampling-based techniques** | Oversampling, cost-sensitive learning | Standard long-tail classification remedies. |
| **No augmentation** | Raw classifier training on imbalanced data | Serves as lower-bound baseline. |

**Evaluation Metrics**

**1. Feature Quality (for tail-class feature hallucination):**

* **Classification Performance**: Accuracy or F1 on tail vs. head vs. overall.
* **Feature Diversity**: Mean pairwise distance or variance within synthetic features.
* **Distributional Match**: KL divergence or MMD between real and synthetic tail distributions.
* **Representation Separation**: Intra-class vs. inter-class cosine similarity (especially head vs tail oncovers overlap).

**2. Image Quality (if using image-level augmentation):**

* **FID / KID** computed separately for tail-class images.
* **CLIP-based Semantic Fidelity**: E.g., CLIP score between generated image and its class prompt.
* **Human Evaluation**: Optional qualitative or expert ratings.

**3. Downstream Impact:**

* **Classification Improvement**: Percent gain over raw baseline and CBDM/CORAL.
* **Few-shot / Long-tail Benchmarks**: Improvements on datasets like CIFAR‑LT, ImageNet‑LT.

**🧠 Integration into Your Framework**

Your memory-conditioned DDPM approach can be **benchmarked directly** against:

* **CBDM** to show benefit of explicit memory conditioning.
* **CORAL** to demonstrate advantage in latent separation.
* **SeedSelect** for prompt-guided baseline comparison.
* **FASA-style augmentation** to contrast simple statistical methods.

By evaluating these baselines and metrics, you'll demonstrate where your algorithm stands in both **feature fidelity/diversity** and **classification downstream performance**.