Compositional Conditioning Consistency Model

賴柏宏 Oral Defense
College of Artificial Intelligence,
National Yang Ming Chiao Tung University
2025 07.02

Outline

1 Introduction

Our Contributions

1. We proposed **CCCM --** the first consistency model capable of compositional zero-shot generation, effectively transferring CCDM's unseen image generation ability into 2–4 step

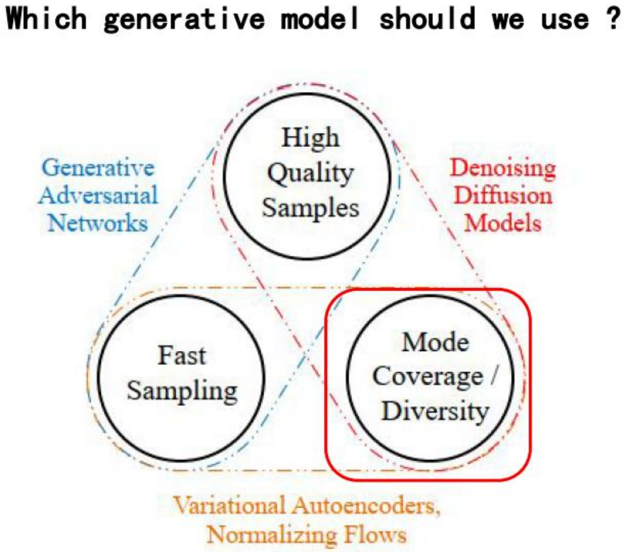
2. **Modified consistency distillation,** combining teacher-predicted and forward-process-formulated supervision. Three fusion strategies: Switch, Step Fuse, and Loss Fuse

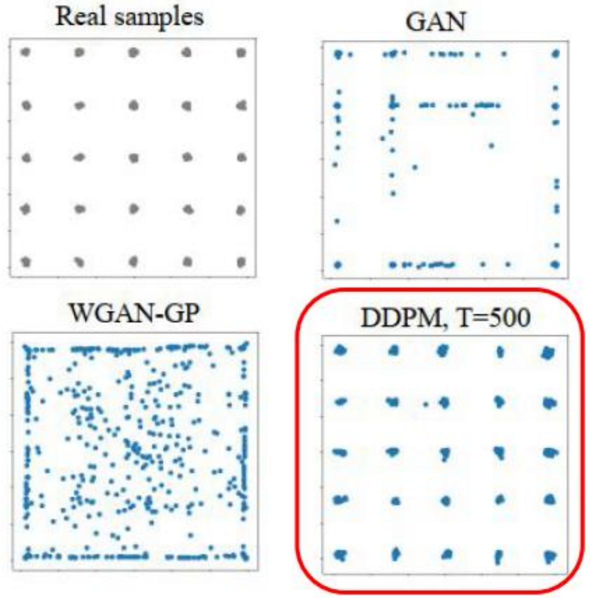
3. CCCM achieves superior FID scores and maintains zero-shot accuracy despite requiring only a fraction of CCDM's sampling steps.

Backgrounds

- Are there neural networks capable of generating unseen classes?

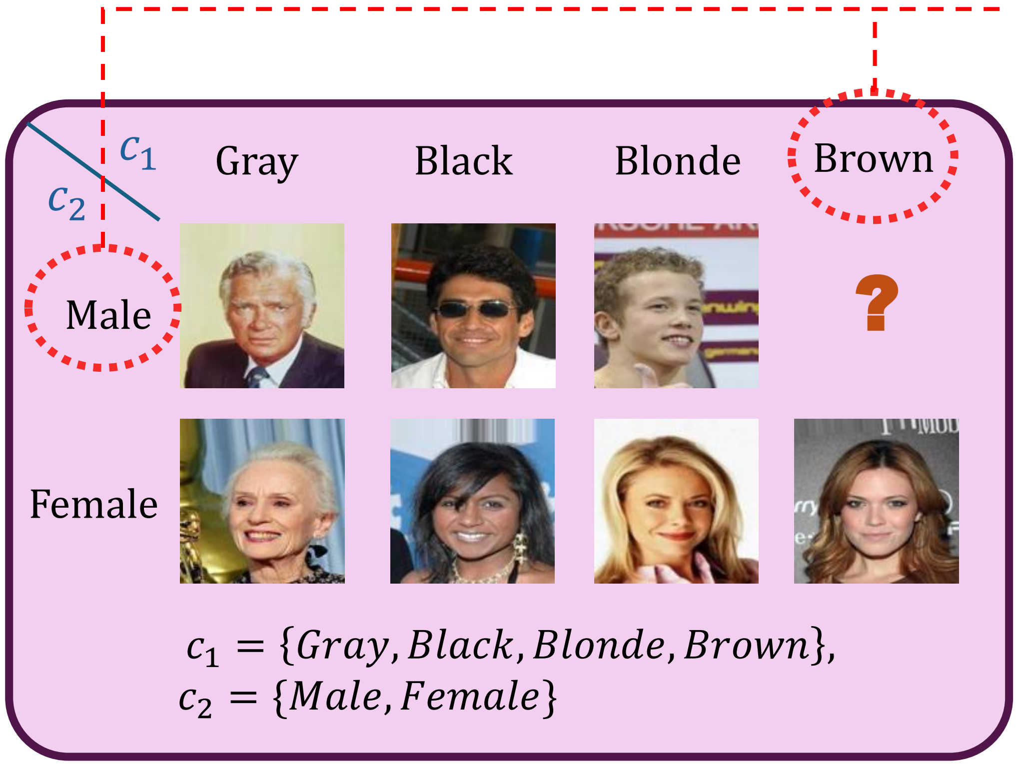
Dataset divided into 8 classes with compositional labels



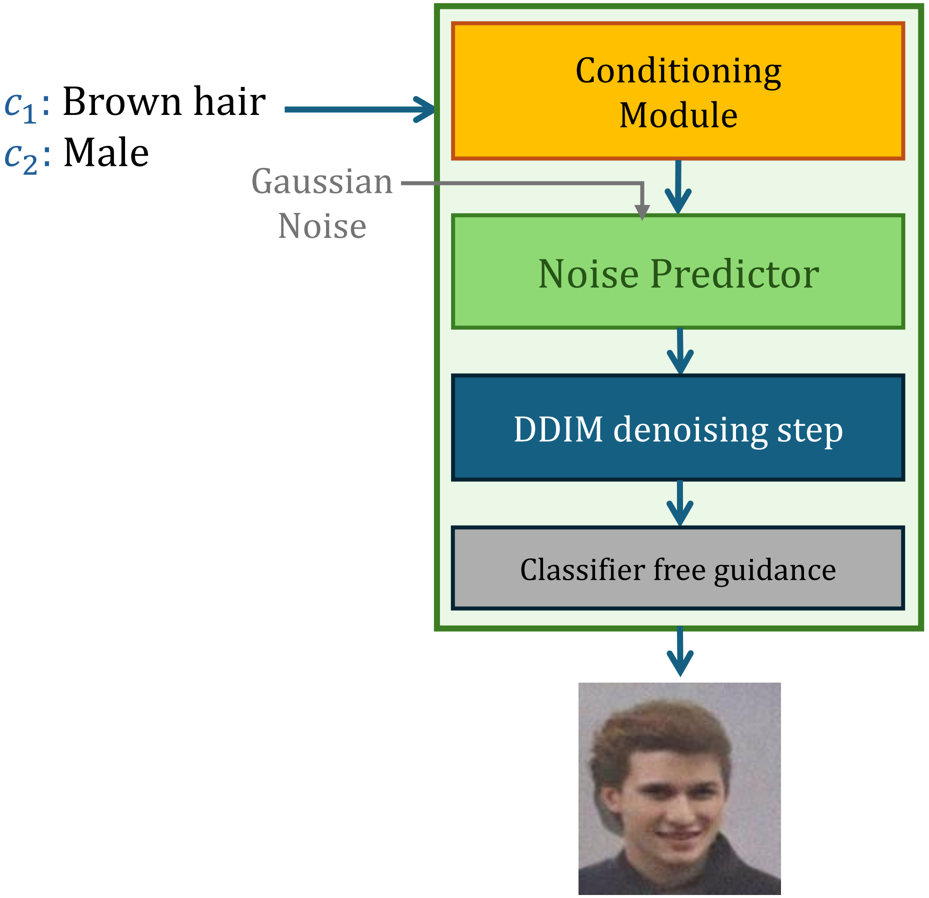


Backgrounds

- **Compositional Conditional Diffusion Models**
  - Capable of generating unseen class images



Compositional Conditional Diffusion Model

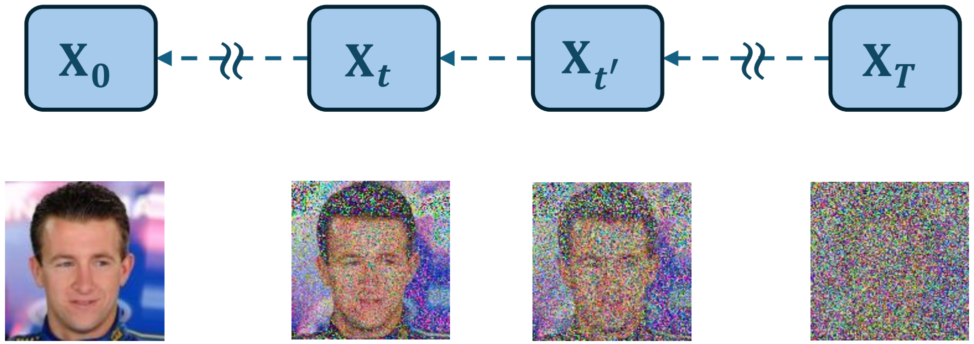


Motivation

- **Can we make it faster (than DDIM) for sampling?**

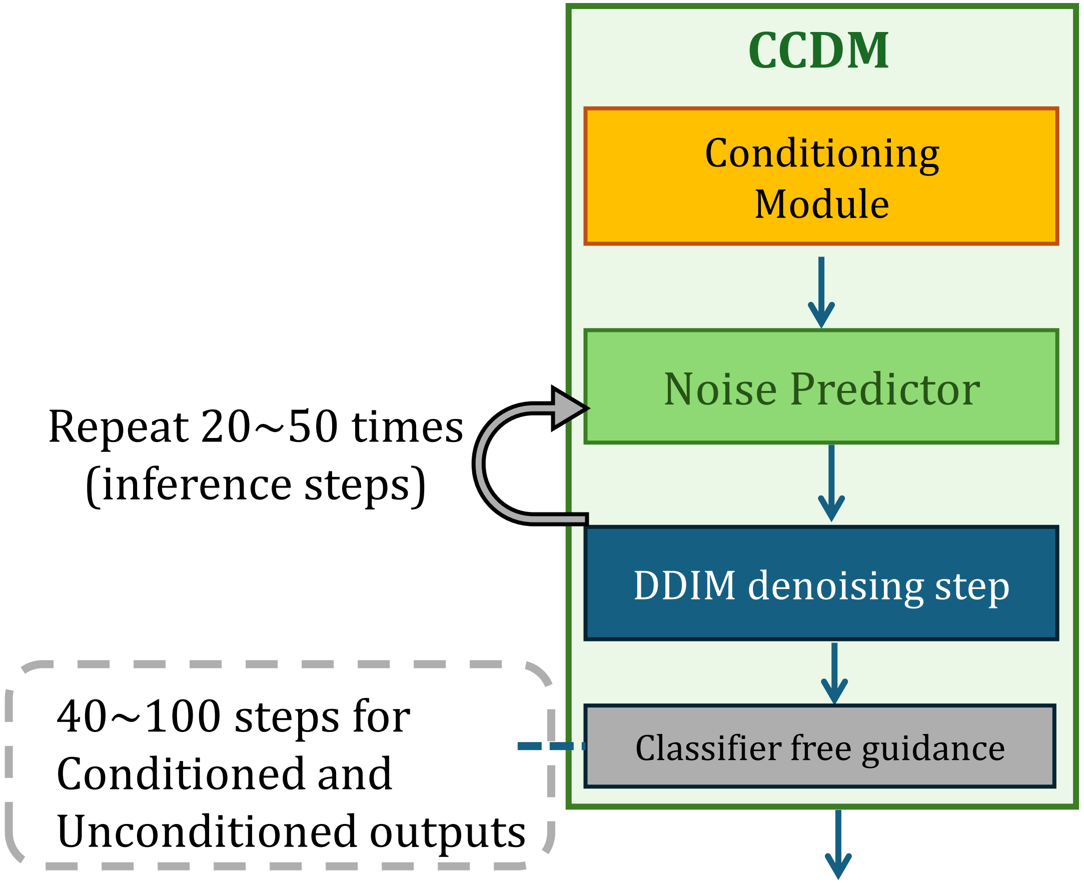
DDIM Reverse Process

20~50 denoising steps!



CFG formulation:

$(1+\omega) * x_0^{Cond} - \omega * x_0^{Uncond}$



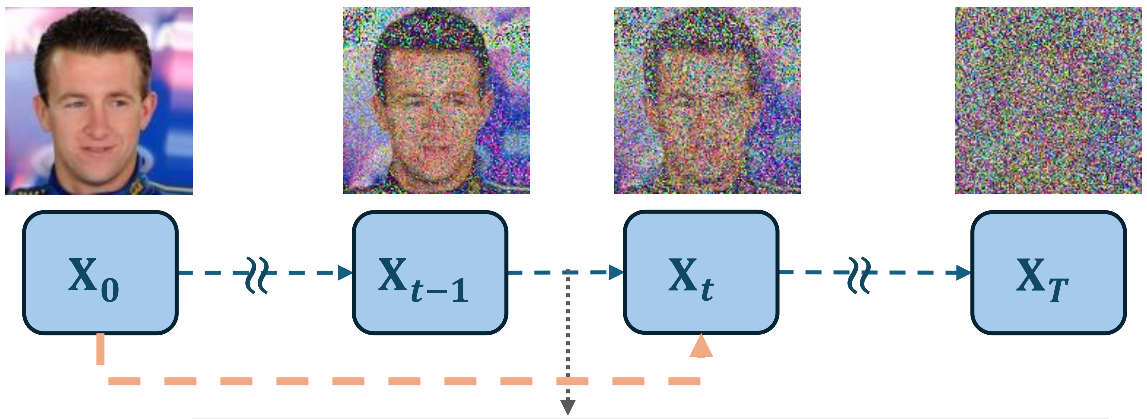
2

Related Works
and
Preliminaries

Diffusion Denoising Probabilistic Models

- Forward diffusion process and reparameterization

Forward Diffusion Process



$$q(x_t \mid x_{t-1}) = \mathcal{N}(x_t ; \sqrt{1-\beta_t} \cdot x_{t-1}, \beta_t \mathbf{I})$$

$$q(x_{1:T} | x_0) = \prod_{t=1}^{T} q(x_t | x_{t-1})$$

$$\bar{\alpha}_t = \prod_{s=1}^{t} (1 - \beta_s)$$

Reparameterization

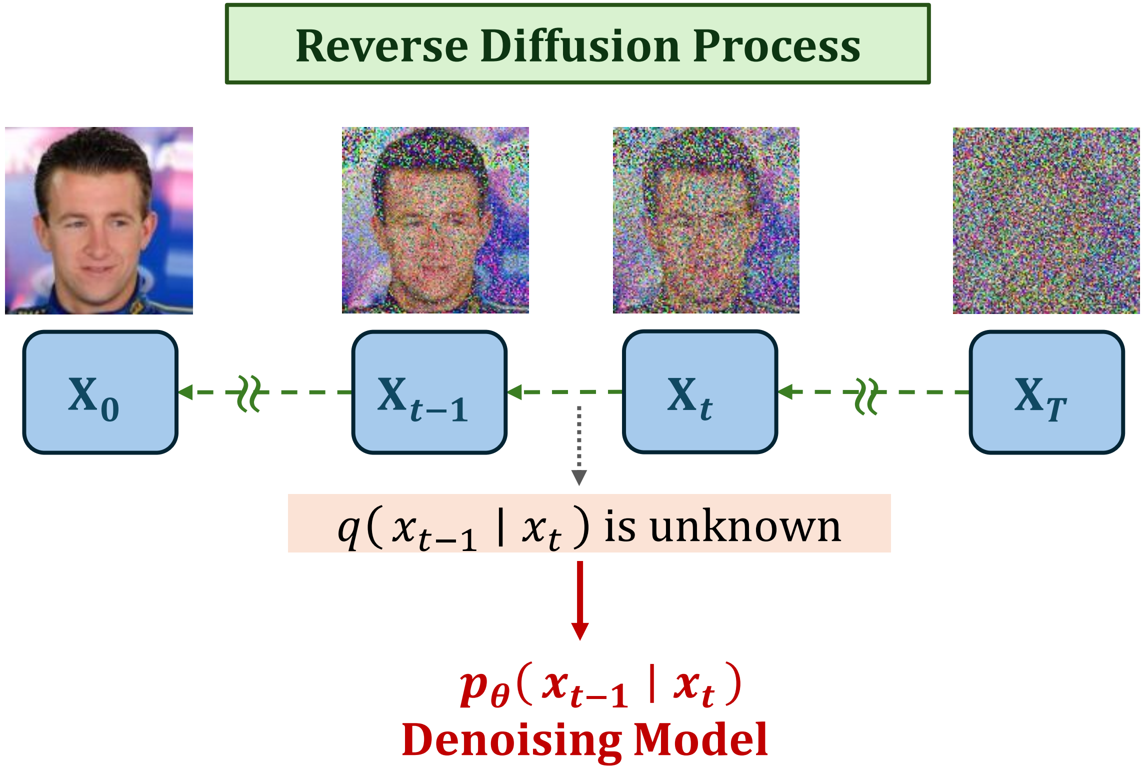
$$q(x_t \mid x_0) = \mathcal{N}\left(x_t ; \sqrt{\bar{\alpha}_t} \cdot x_0, (1-\bar{\alpha}_t) \cdot \mathbf{I}\right)$$

forward process formula

$$x_t = \sqrt{\bar{\alpha}_t} \cdot x_0 + \sqrt{1-\bar{\alpha}_t} \cdot \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \mathbf{I})$$

Diffusion Denoising Probabilistic Models

- Reverse diffusion process, training and sampling



Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
MSE loss $\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \ldots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t)\right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

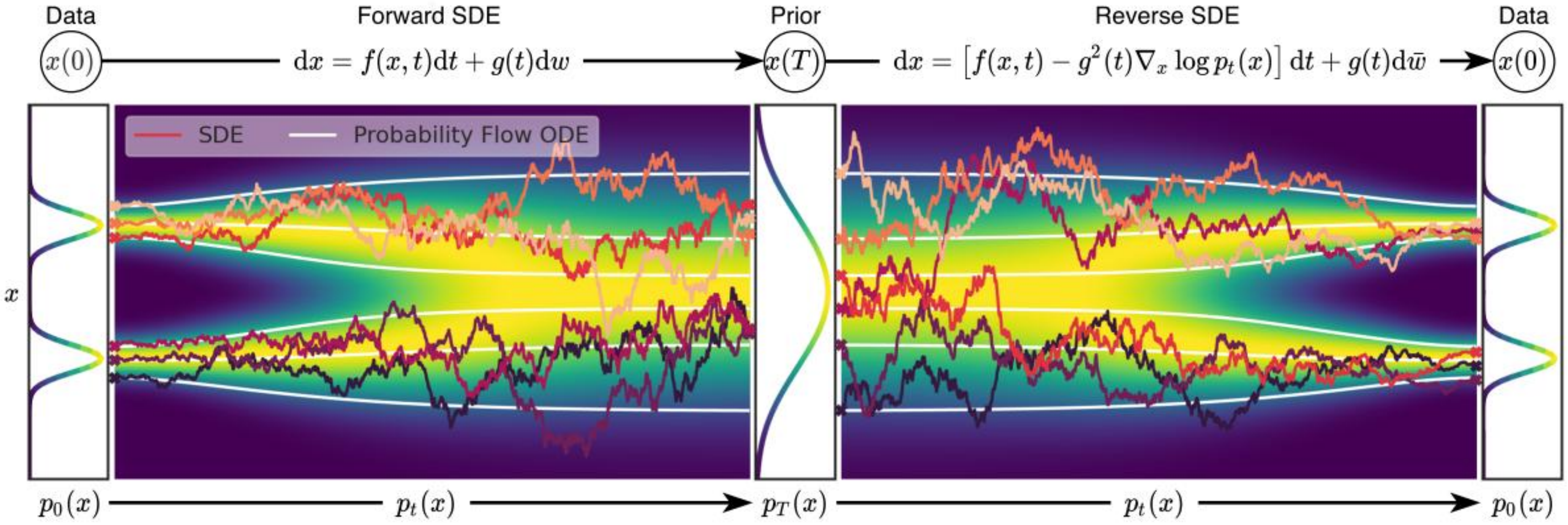
stochasticity

Score-Based Generative Modeling Through SDE

- Stochastic differential equation & Ordinary differential equation

Reverse ODE

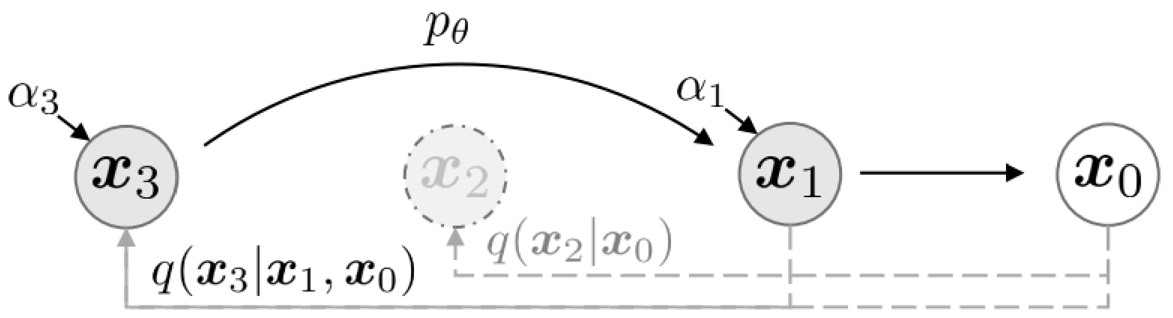
$$dx = [f(x,t) - \frac{1}{2} g(t)^2 \nabla_x \log p_t(x)]dt$$



"For all diffusion processes, there exists a corresponding *deterministic process*, whose trajectories share the same marginal probability densities as the SDE."

Diffusion Denoising Implicit Models

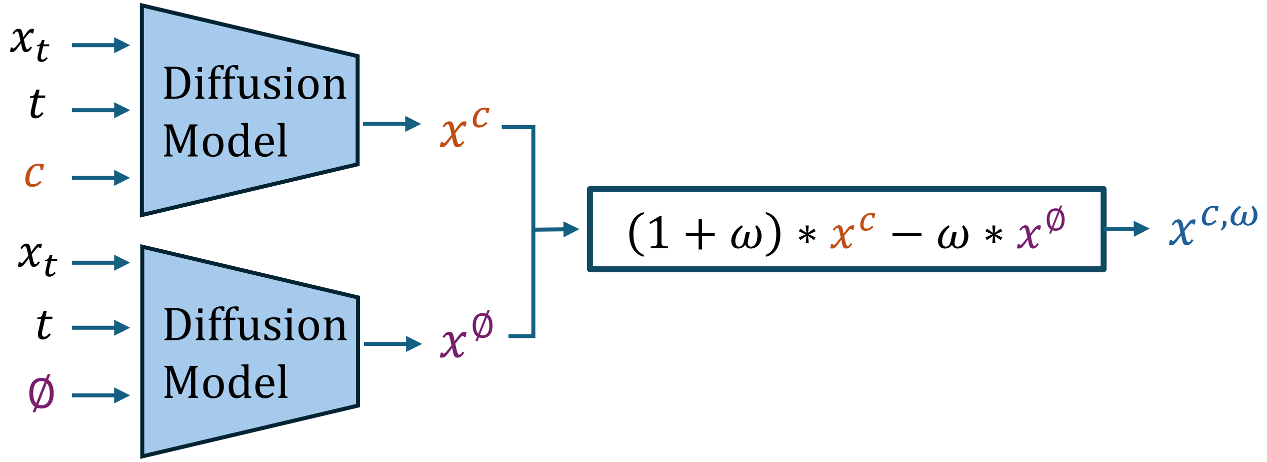
- Non-Markovian, deterministic denoising
- Speeds up sampling
- No **retraining** required to utilize DDIM sampling



$$\boldsymbol{x}_{t-1} = \sqrt{\alpha_{t-1}} \underbrace{\left(\frac{\boldsymbol{x}_t - \sqrt{1-\alpha_t}\,\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}{\sqrt{\alpha_t}}\right)}_{\text{“ predicted } \boldsymbol{x}_0\text{”}} + \underbrace{\sqrt{1-\alpha_{t-1}-\cancel{\sigma_t^2}}\cdot\epsilon_\theta^{(t)}(\boldsymbol{x}_t)}_{\text{“direction pointing to } \boldsymbol{x}_t\text{”}} + \underbrace{\cancel{\sigma_t\epsilon_t}}_{\text{random noise}}$$

Classifier free guidance

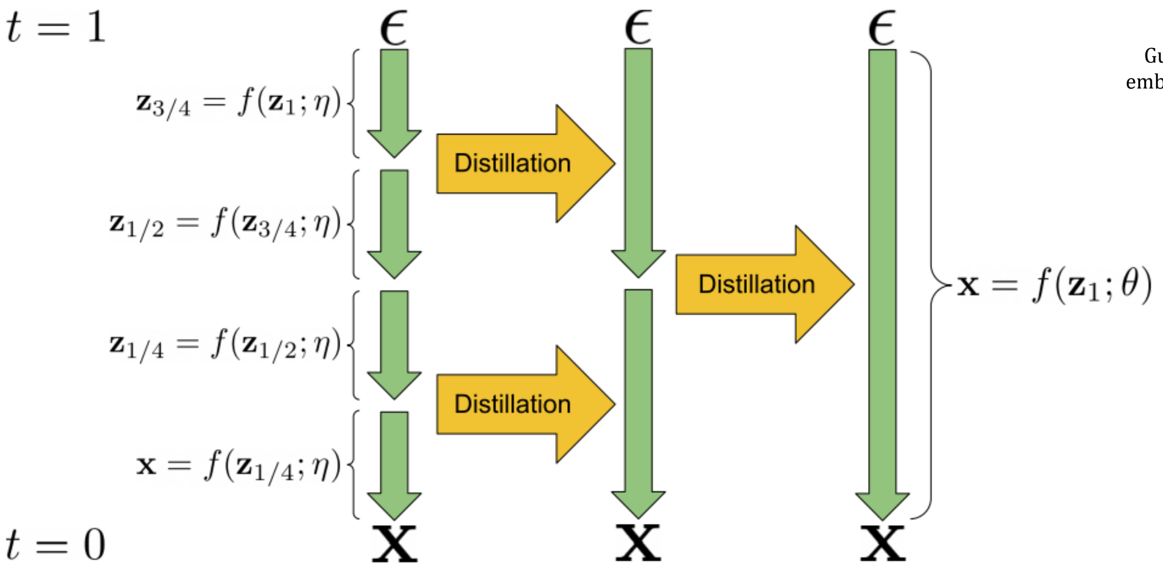
- **Doesn't** need **another classifier** to guide the Diffusion model.
- When training, mostly learns **conditioned** output x^c with input c.
- Occasionally **drops** condition for **unconditioned** output $x^\emptyset$.
- Weighted combination of conditioned output and unconditioned output.



➢ Problem: Needs **twice** (cond+uncond) as **many inference steps.**

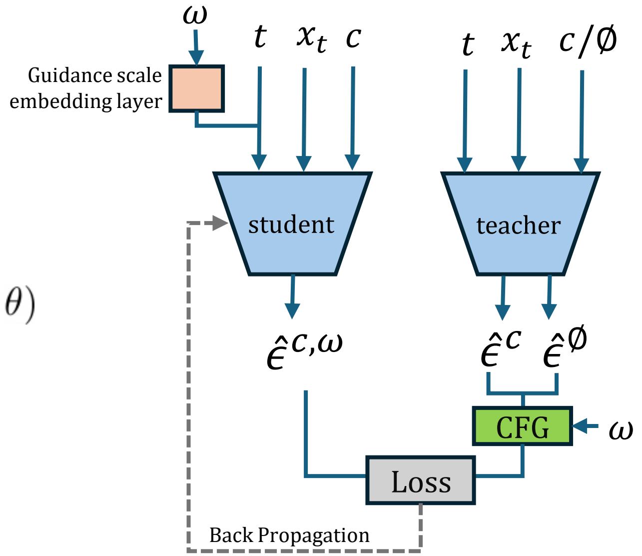
Distillation on Diffusion models

Student from previous round as the new Teacher

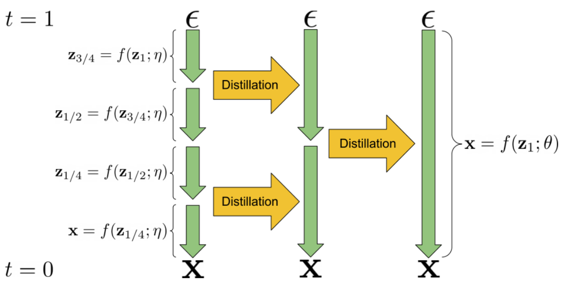


Progressive Distillation [6]

Stage 1-
eliminate CFG



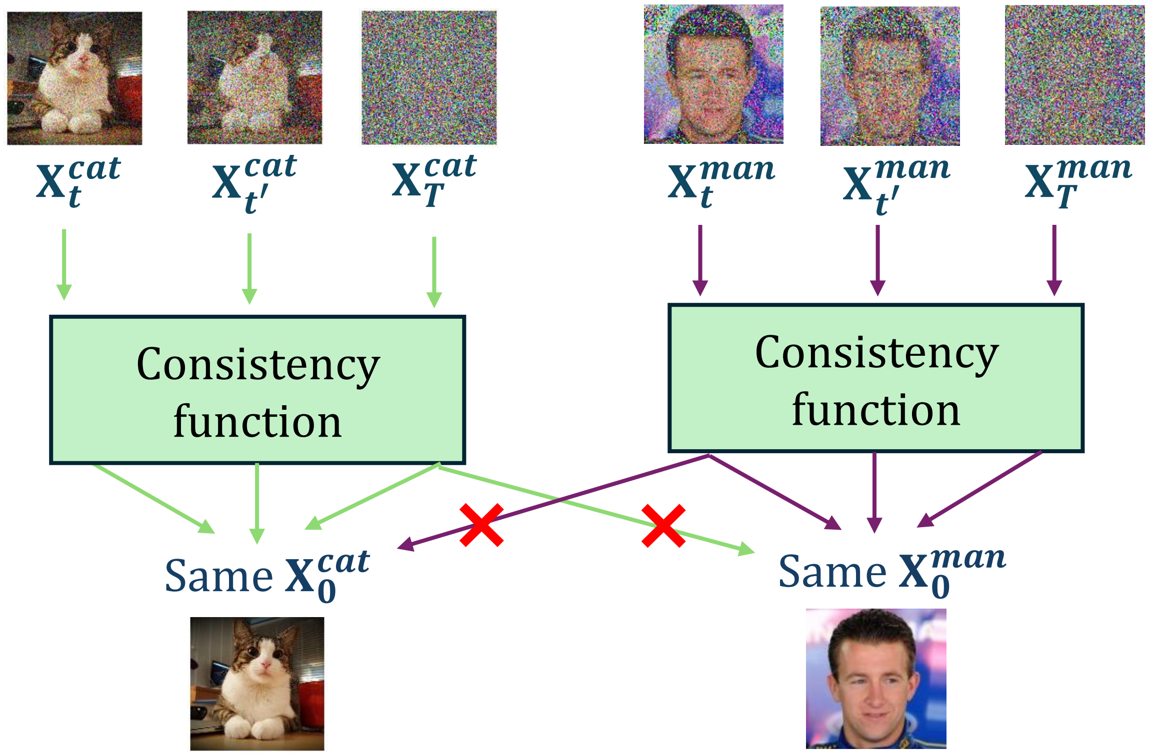
Stage 2-
Progressive Distillation



2-stage Distillation on Guided Diffusion models[7]

Consistency Models

- Consistency function
  - Given any $t \in [0, .. T]$, exists a function: $f(x_t) \rightarrow x_0$
  - Same x_0 for $x_t, x_{t'}, x_T$ on the same diffusion trajectory



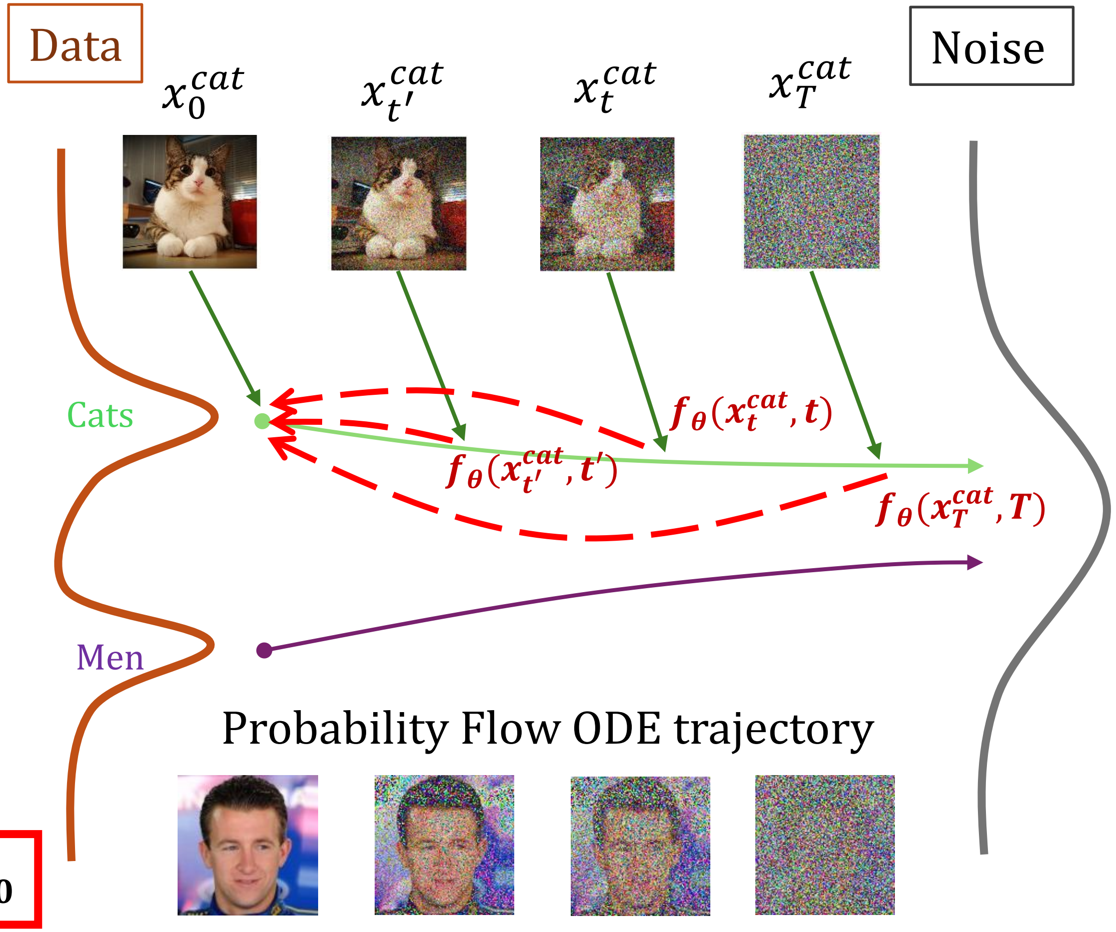
Consistency Models

- Loss function for training
  - Train a **model $\boldsymbol{f_\theta}$** to approximate this function
  - Due to the **deterministic** property of the **PF-ODE**, we may train a consistency model by enforcing the **Consistency objective**
  - Loss function for training:

$$\min_{\theta}[\, d\left(f_\theta(x_{t_n}, t_n), f_\theta\left(x_{t_{n-1}}, t_{n-1}\right)\right)],$$

So that :

$$f_\theta(x_{t_n}, t_n) = f_\theta(x_{t_{n-1}}, t_{n-1}) = f_\theta(x_{t_1}, t_1) = f_\theta(x_0, t_0) = \boxed{\hat{\boldsymbol{x}}_0}$$



Consistency Models

- Consistency Distillation
  - How to obtain adjacent point $x_{t_{n-1}}$?
    - May use forward process formula
    - **Or Better –** Teacher model f_{ϕ} as the ODE solver $\mathbf{\Phi}$
  - **Consistency Distillation**

Distill a pretrained Diffusion model into a CM

Algorithm 2 Consistency Distillation (CD)

Input: dataset $\mathcal{D}$, initial model parameter $\boldsymbol{\theta}$, learning rate η, ODE solver $\Phi(\cdot,\cdot;\phi)$, $d(\cdot,\cdot)$, $\lambda(\cdot)$, and μ

$\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$

repeat

Sample $\mathbf{x} \sim \mathcal{D}$ and $n \sim \mathcal{U}[\![1, N-1]\!]$

Sample $\mathbf{x}_{t_{n+1}} \sim \mathcal{N}(\mathbf{x}; t_{n+1}^2 \boldsymbol{I})$

$\hat{\mathbf{x}}_{t_n}^{\phi} \leftarrow \mathbf{x}_{t_{n+1}} + (t_n - t_{n+1})\Phi(\mathbf{x}_{t_{n+1}}, t_{n+1}; \phi)$

$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \phi) \leftarrow$

$\lambda(t_n) d(\boldsymbol{f}_{\boldsymbol{\theta}}(\mathbf{x}_{t_{n+1}}, t_{n+1}), \boldsymbol{f}_{\boldsymbol{\theta}^-}(\hat{\mathbf{x}}_{t_n}^{\phi}, t_n))$

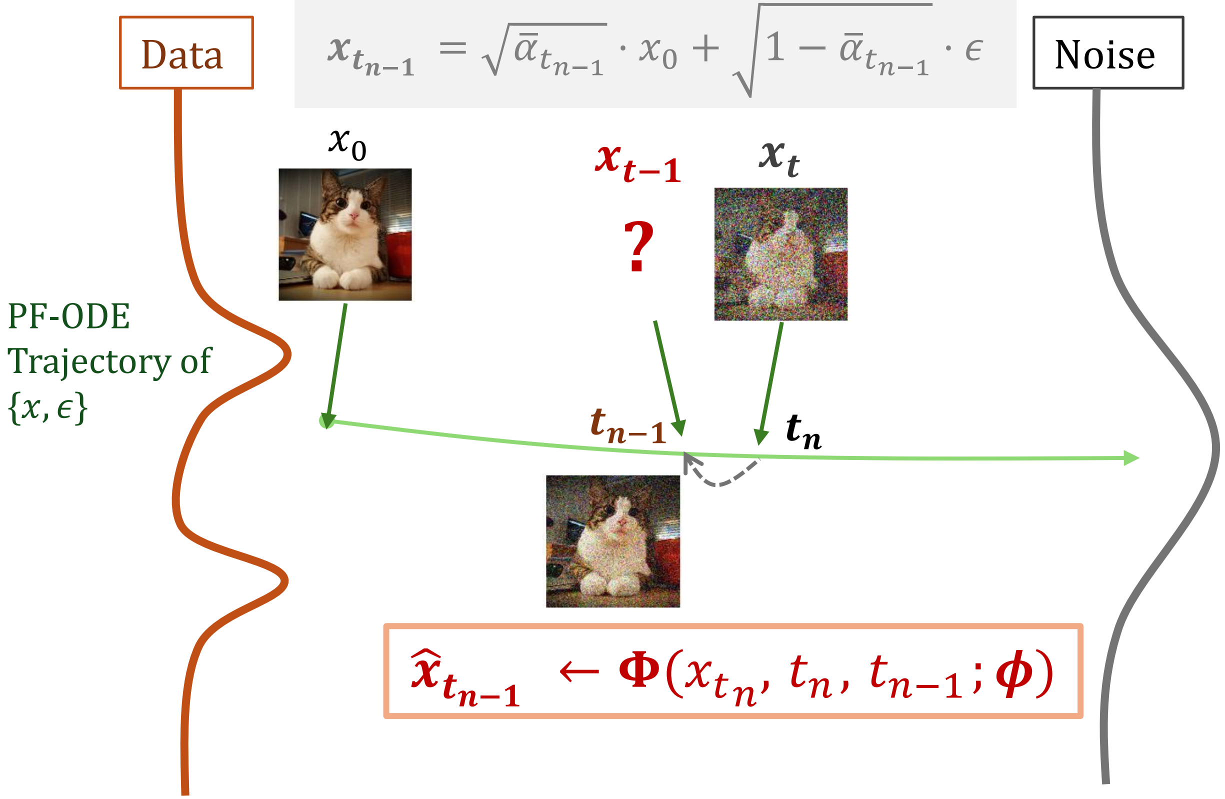
$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\theta}^-; \phi)$

$\boldsymbol{\theta}^- \leftarrow \text{stopgrad}(\mu \boldsymbol{\theta}^- + (1-\mu)\boldsymbol{\theta})$

until convergence

$$x_{t_n} = \sqrt{\bar{\alpha}_{t_n}} \cdot x_0 + \sqrt{1 - \bar{\alpha}_{t_n}} \cdot \epsilon$$

$$x_{t_{n-1}} = \sqrt{\bar{\alpha}_{t_{n-1}}} \cdot x_0 + \sqrt{1 - \bar{\alpha}_{t_{n-1}}} \cdot \epsilon$$



Adjacent point on PF-ODE trajectory

Consistency Models

- Sampling

Single step or Multistep Consistency Sampling

Algorithm 1 Multistep Consistency Sampling

Input: Consistency model $\boldsymbol{f}_{\boldsymbol{\theta}}(\cdot,\cdot)$, sequence of time points $\tau_1 > \tau_2 > \cdots > \tau_{N-1}$, initial noise $\hat{\mathbf{x}}_T$

$\mathbf{x} \leftarrow \boldsymbol{f}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_T, T)$

for $n = 1$ **to** $N-1$ **do**

  Sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$

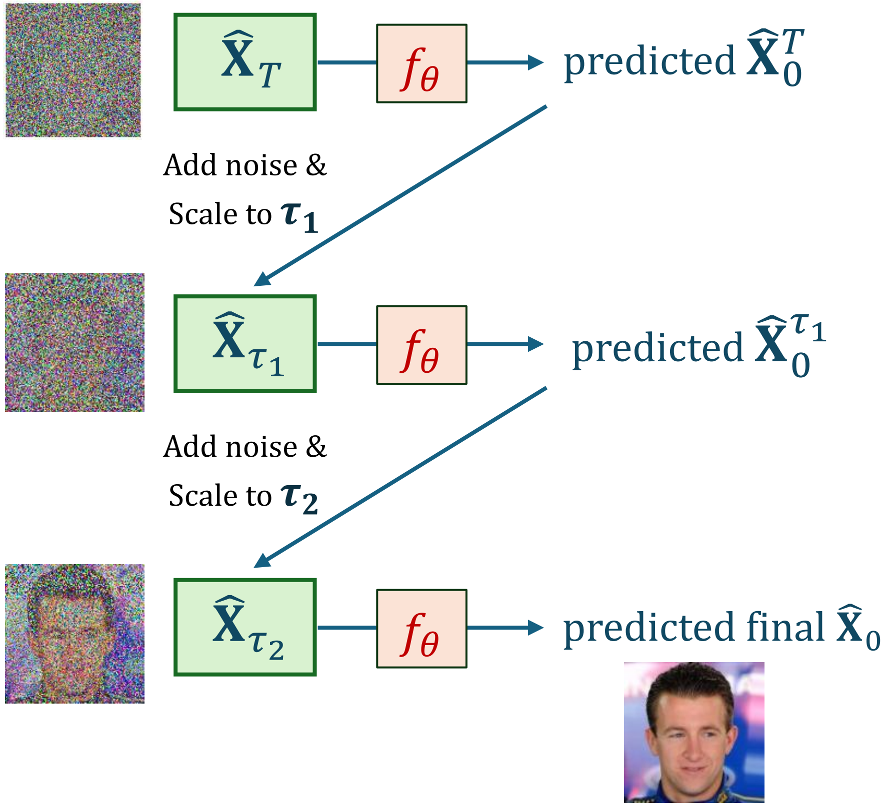
  $\hat{\mathbf{x}}_{\tau_n} \leftarrow \mathbf{x} + \sqrt{\tau_n^2 - \epsilon^2}\,\mathbf{z}$

  $\mathbf{x} \leftarrow \boldsymbol{f}_{\boldsymbol{\theta}}(\hat{\mathbf{x}}_{\tau_n}, \tau_n)$

end for

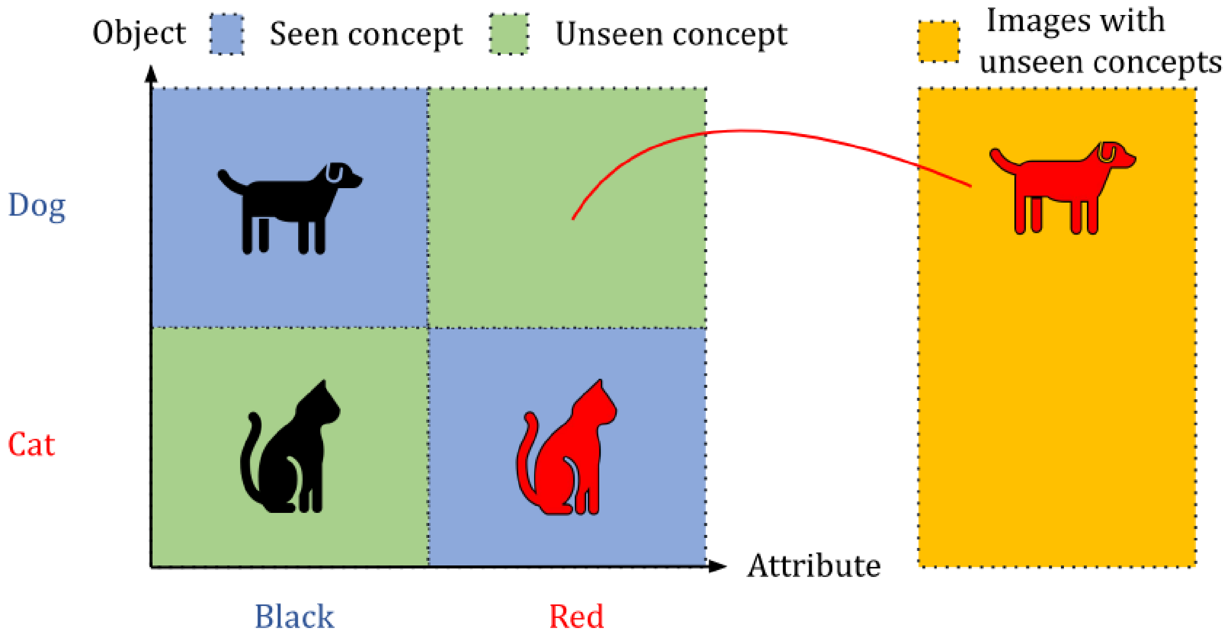
Output: $\mathbf{x}$

Example diagram of 3 step sampling

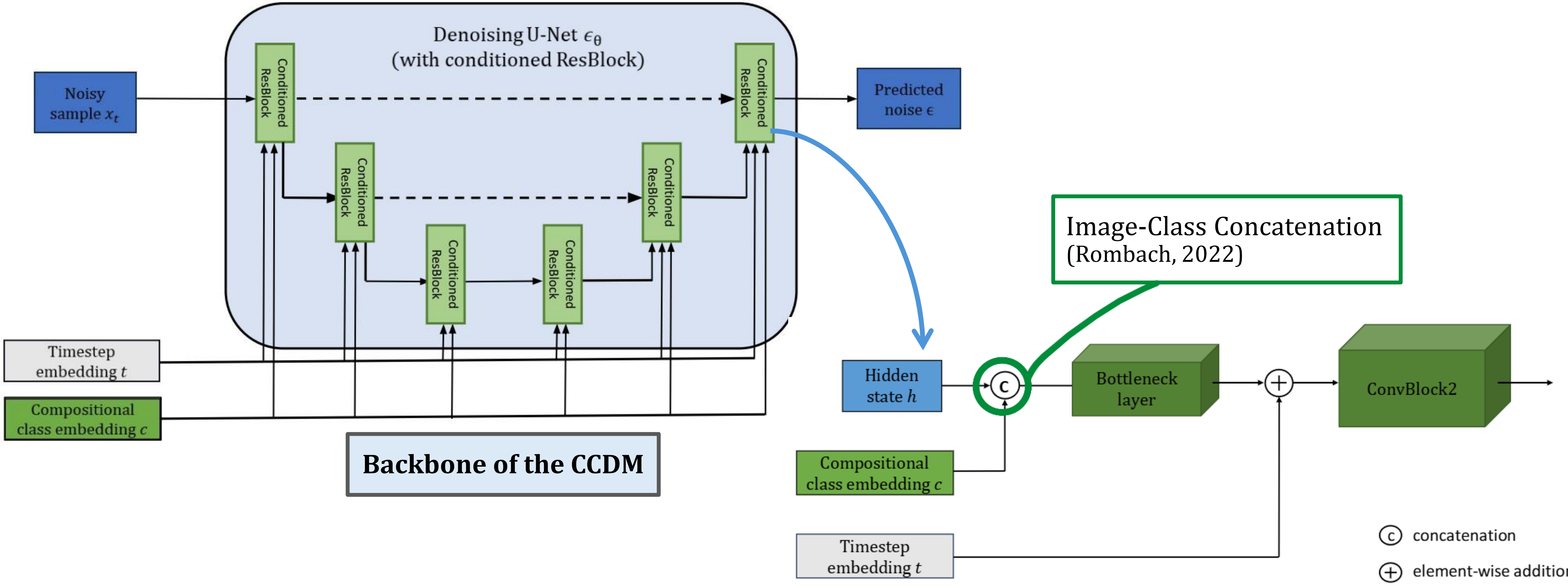


Compositional Zero-Shot Learning

- CZSL allows models to predict unseen classes by leveraging a combination of zero-shot learning and compositional understanding.



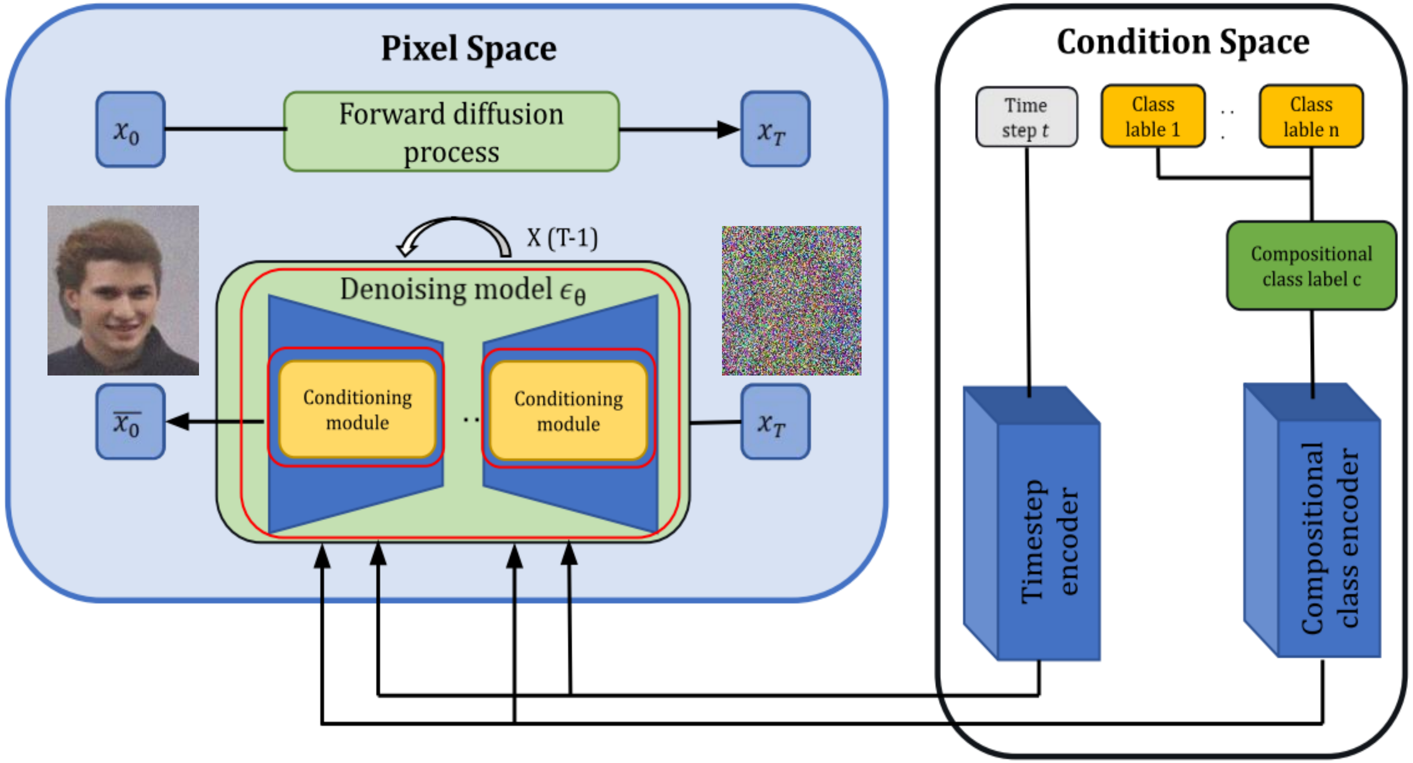
Compositional Conditional Diffusion Models



Backbone of the CCDM

Image-Class Concatenation
(Rombach, 2022)

Compositional Conditional Diffusion Models



Algorithm Training CCDM

1: **repeat**:
2: $(x_0, c) \sim p(x, c)$
3: $c \leftarrow \varnothing$ with probability p_{uncond}
4: $t \sim Uniform(\{1, \ldots, T\})$
5: $\epsilon \sim \mathcal{N}(0, \mathbf{I})$
6: $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon$
7: Take a gradient step on $\nabla_\theta \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} \epsilon, t, c)\|^2$
8: **until** converged

3

Proposed Method

Compositional Conditional Consistency Model

- Using CCDM as the teacher model for consistency distillation

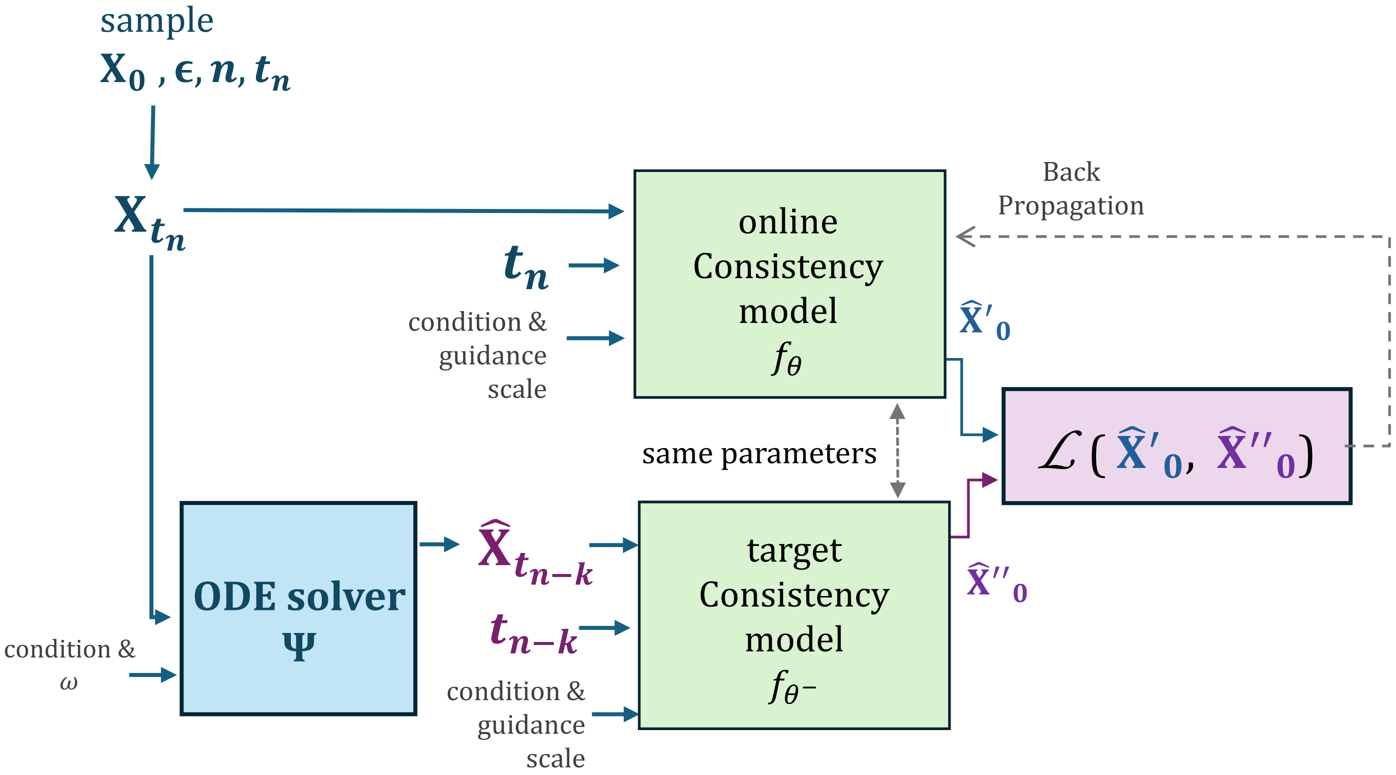
Follows Latent Consistency Model's (Luo, 2023) implementation of :

- skipping step $\boldsymbol{k}$ =20

  Reduces training time

- guidance scale embedding layer

High level block diagram of compositional consistency distillation

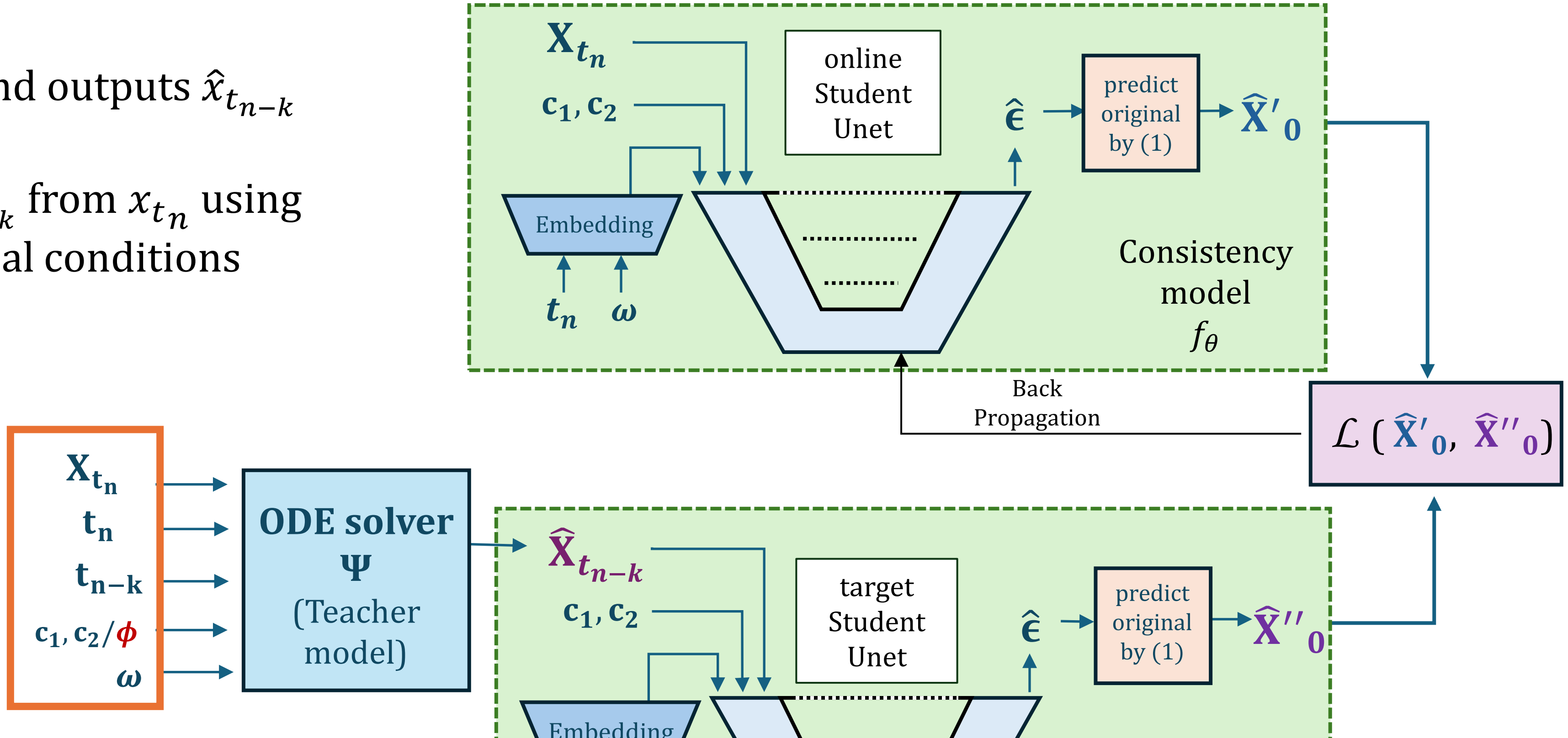


Notations:
$n \in [1,2,..T]$: step index
$t_n \in [0,1]$: time at step n
$\epsilon \sim \mathcal{N}(0, \mathbf{I})$: Gaussian Noise
X_{t_n}: noisy image at t_n
$\hat{\mathrm{X}}_{t_n}$: predicted image at t_n
X_0: clean image from dataset
$\hat{\mathrm{X}}_0$: predicted clean image
ω : guidance scale

Compositional Conditional Consistency Model

- Detailed block diagram of compositional consistency distillation

  - ODE solver takes input x_{t_n} and outputs $\hat{x}_{t_{n-k}}$
  - Deterministically estimates $\hat{x}_{t_{n-k}}$ from x_{t_n} using CCDM + DDIM under compositional conditions

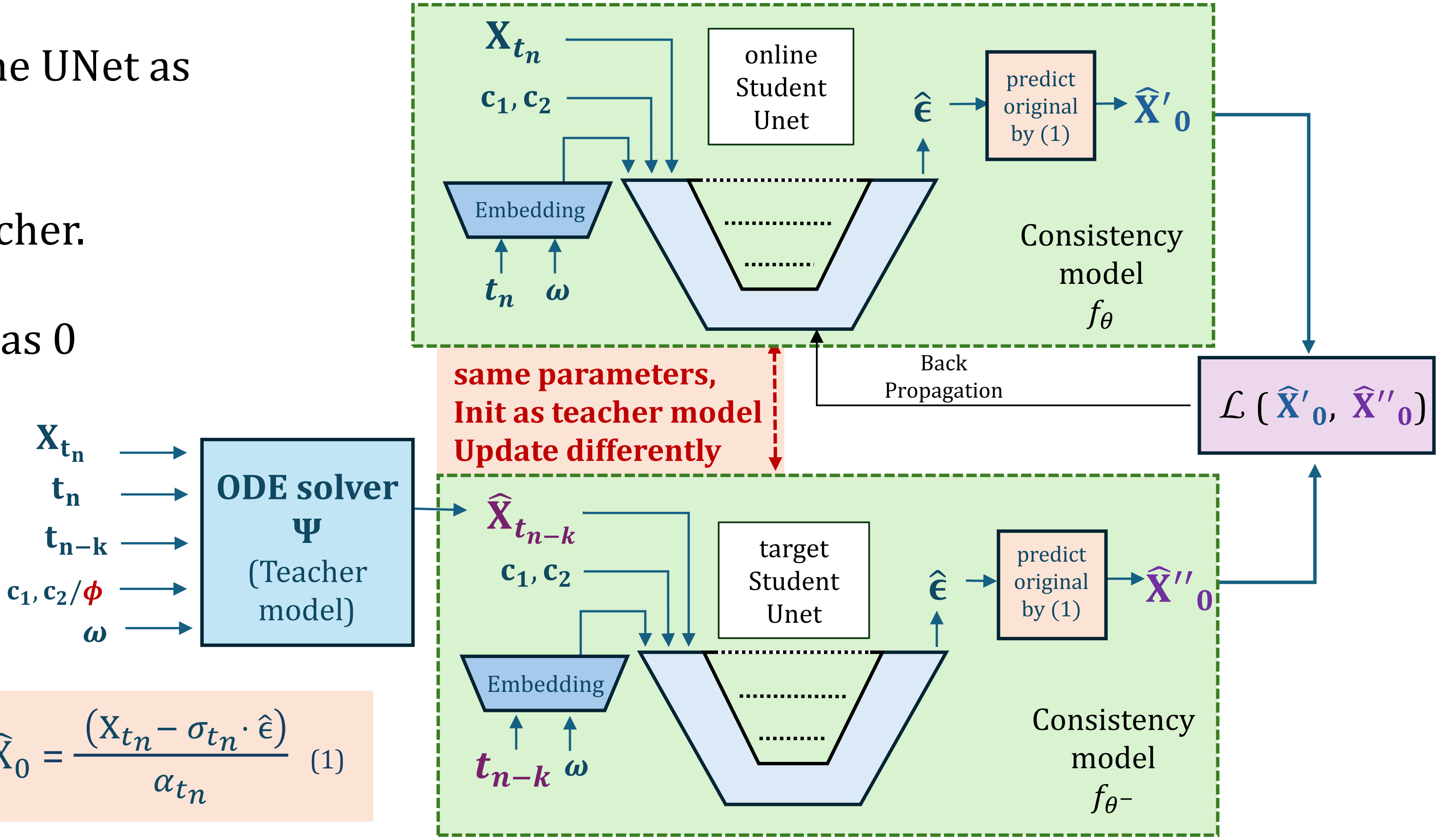


Notations:
- $n \in [1,2,..T]$: step index
- $t_n \in [0,1]$: time at step n
- X_{t_n}: noisy image at t_n
- $\hat{X}_{t_n}$: predicted image at t_n
- X_0 : clean image from dataset
- $\hat{X}_0$: predicted clean image
- α_{t_n} :signal rate at t_n
- σ_{t_n} :noise rate at t_n
- $\hat{\epsilon}$: predicted noise
- ω : guidance scale

$$\hat{X}_0 = \frac{(X_{t_n} - \sigma_{t_n} \cdot \hat{\epsilon})}{\alpha_{t_n}} \quad (1)$$

Compositional Conditional Consistency Model

- Detailed block diagram of compositional consistency distillation
  - f_θ and f_{θ^-} using same backbone UNet as teacher (CCDM)
  - both weights initialized as teacher.
  - ω embedding layer initialized as 0

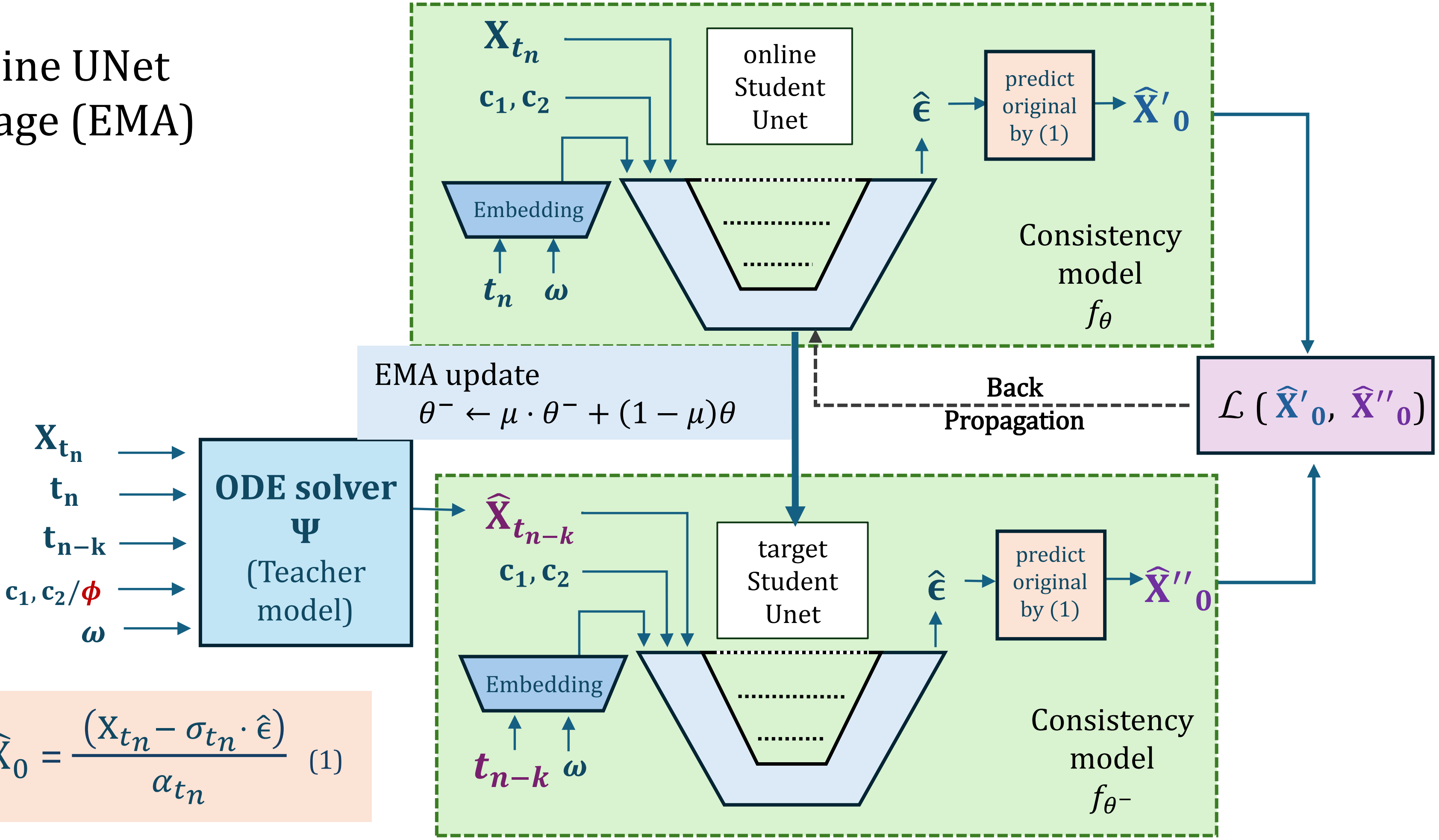


Notations:
$n \in [1,2,..T]$: step index
$t_n \in [0,1]$: time at step n
X_{t_n}: noisy image at t_n
$\hat{X}_{t_n}$: predicted image at t_n
X_0 : clean image from dataset
$\hat{X}_0$: predicted clean image
α_{t_n} :signal rate at t_n
σ_{t_n} :noise rate at t_n
$\hat{\epsilon}$: predicted noise
ω : guidance scale

$$\hat{X}_0 = \frac{(X_{t_n} - \sigma_{t_n} \cdot \hat{\epsilon})}{\alpha_{t_n}} \quad (1)$$

Compositional Conditional Consistency Model

- Detailed block diagram of compositional consistency distillation
  - Target UNet updated from online UNet using Exponential Moving Average (EMA)



Notations:
- $n \in [1,2,..T]$: step index
- $t_n \in [0,1]$: time at step n
- X_{t_n}: noisy image at t_n
- $\hat{X}_{t_n}$: predicted image at t_n
- X_0 : clean image from dataset
- $\hat{X}_0$: predicted clean image
- α_{t_n} :signal rate at t_n
- σ_{t_n} :noise rate at t_n
- $\hat{\epsilon}$: predicted noise
- ω : guidance scale

$$\hat{X}_0 = \frac{(X_{t_n} - \sigma_{t_n} \cdot \hat{\epsilon})}{\alpha_{t_n}} \quad (1)$$

Ways of Computing $X_{t_{n-k}}$

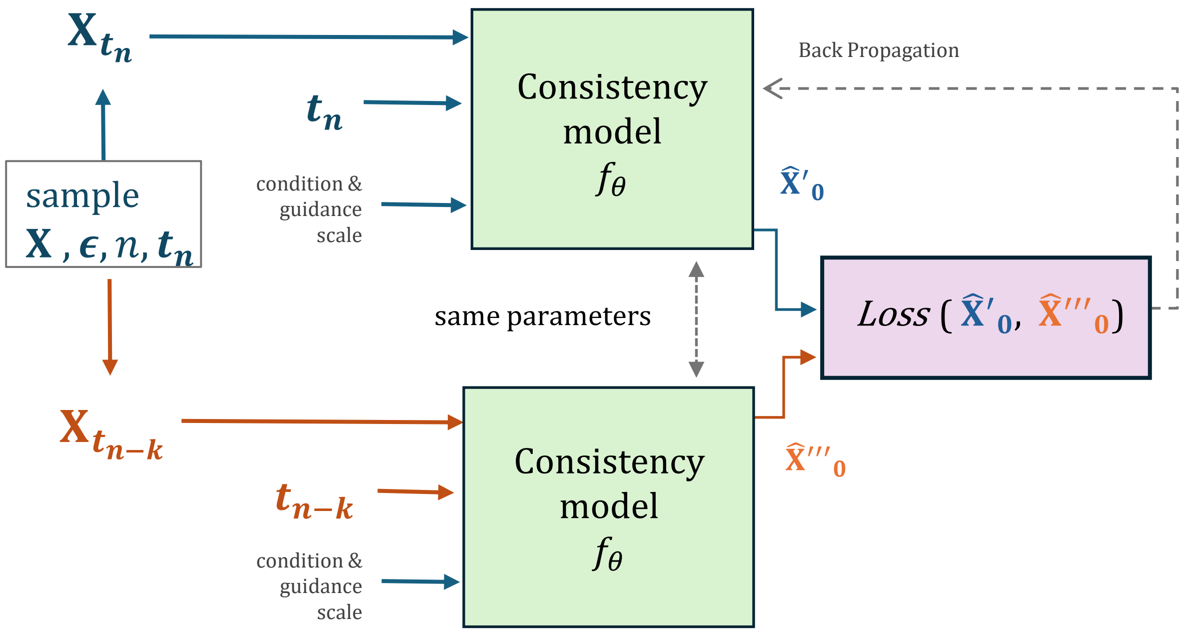
- Teacher predicted or Forward-process formulation?

  - CCCM with teacher's supervision offers high-quality samples.

  - Could formulated $x_{t_{n-k}}$ do better?
    - ➢ Not quite if solely rely on formulated.

$$x_{t_{n-k}} = \sqrt{\bar{\alpha}_{t_{n-k}}} \cdot x_0 + \sqrt{1-\bar{\alpha}_{t_{n-k}}} \cdot \epsilon, \epsilon \text{ same as in } x_{t_n}$$

  - Gradually shifting from teacher to formulated $x_{t_{n-k}}$ might help?



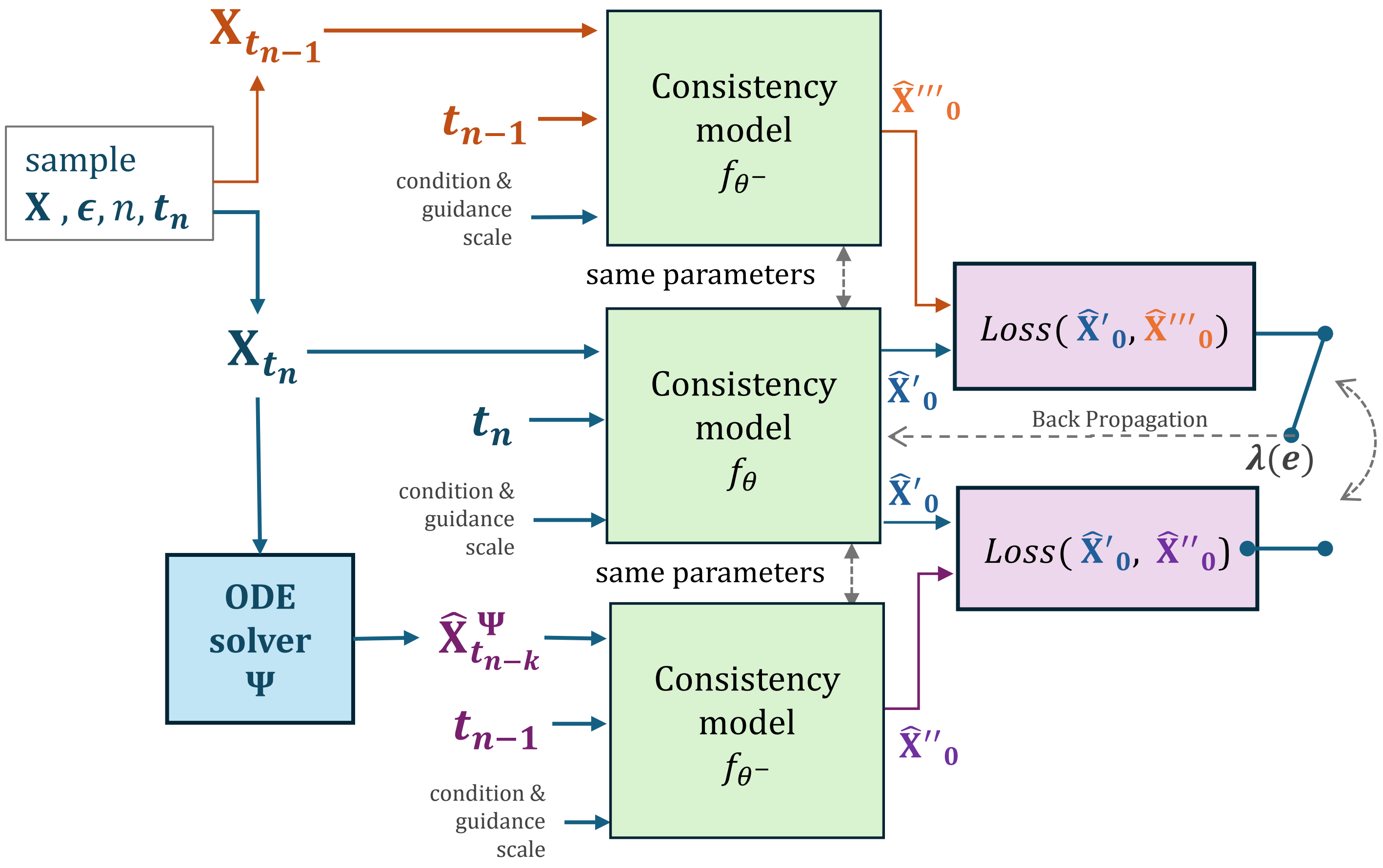
Epoch function $\lambda(e)$- Fusion implementation

- To control the strength of 2 branches: $\hat{x}_{t_{n-k}}$ and $x_{t_{n-k}}$
  - Epoch function $\lambda(e): \{1,2,\dots,e_{max}\} \rightarrow [0,1]$.
    1 = fully teacher signal, 0 = formulated signal
  - Switch, Step Fuse, and Loss Fuse implemented via $\lambda(e)$
  - Controlled by Fuse Scheduler

Modified Consistency Distillation

- Switch Strategy
  - Switches source of $x_{t_{n-k}}$ based on an epoch threshold :

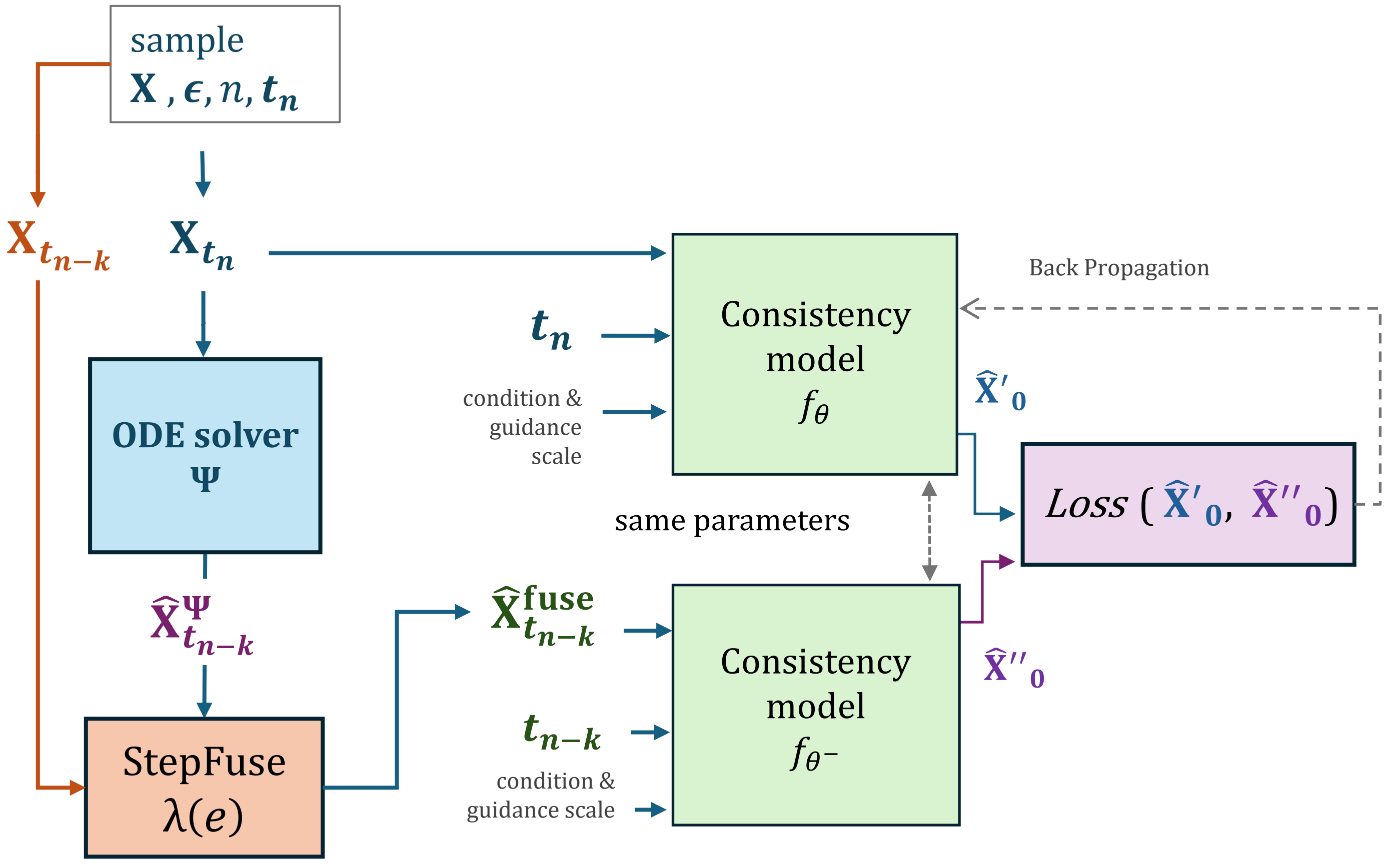
$$\lambda(e) = \begin{cases} 1, & e < threshold \\ 0, & e \geq threshold \end{cases}$$



Modified Consistency Distillation

- Step-Fuse Strategy
  - Pixel-wise weighted sum of two x_{t-k} sources:

$$\hat{x}_{t_{n-k}}^{\text{fuse}} = \lambda(e) \cdot \hat{x}_{t_{n-k}}^{\Psi} + (1 - \lambda(e)) \cdot x_{\text{t}-\text{k}}$$



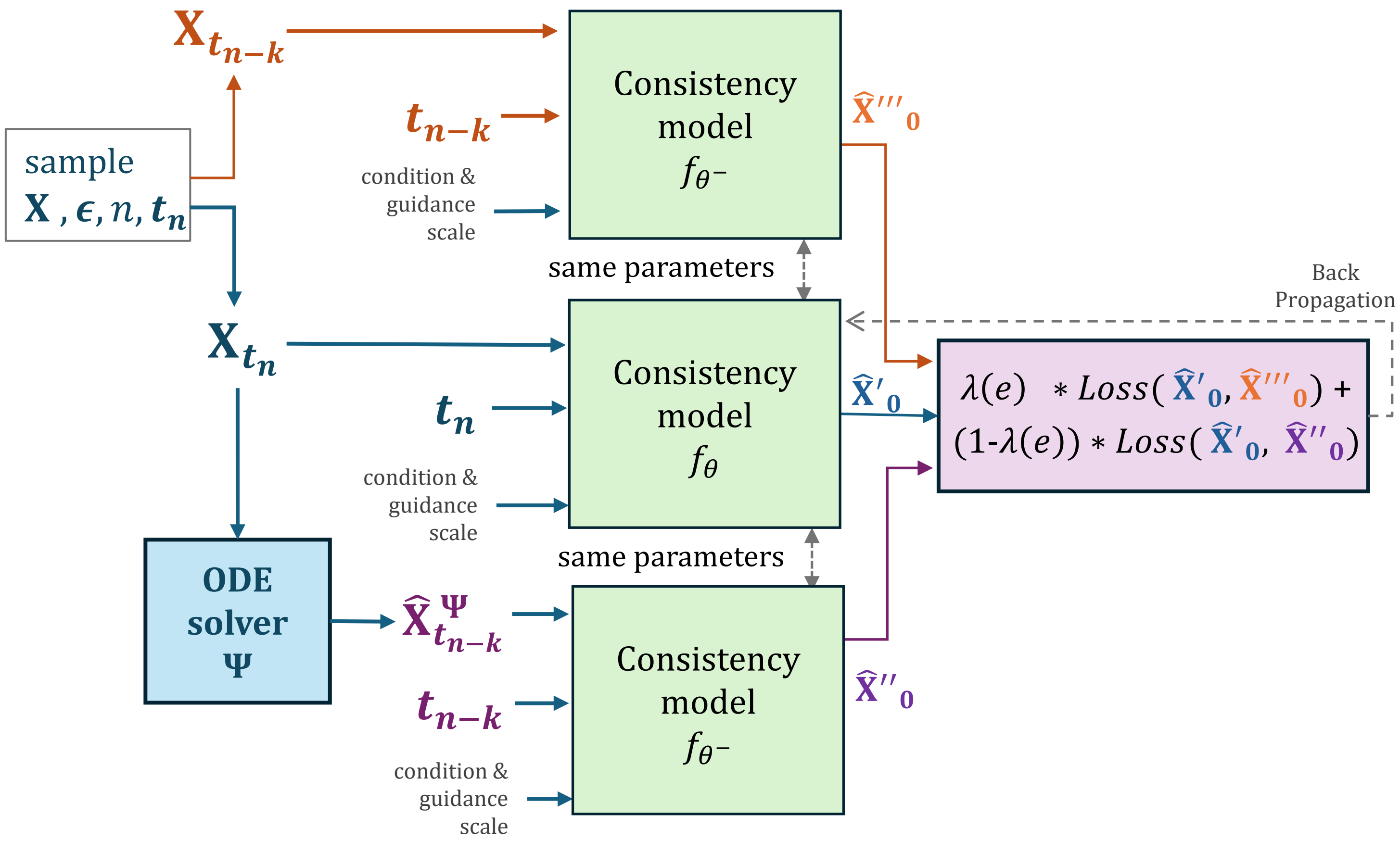
Modified Consistency Distillation

- Loss-Fuse Strategy
  - Weighted sum of two loss terms:

$$\mathcal{L}_{teacher} = \mathbb{E}\left[d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega), f_\theta(\hat{x}^{\Psi}_{t_{n-k}}, t_{n-k}, c_1, c_2, \omega)\right)\right]$$

$$\mathcal{L}_{formulated} = \mathbb{E}\left[d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega), f_\theta(x_{t_{n-k}}, t_{n-k}, c_1, c_2, \omega)\right)\right]$$

$$loss = \lambda(e) \cdot \mathcal{L}_{teacher} + (1 - \lambda(e)) \cdot \mathcal{L}_{formulated}$$



Fuse Scheduler design

- Constant:

  $\lambda(epoch) = \lambda_0$,

  fixed blending weight throughout training.

- Exponential Decay:

  $\lambda(epoch) = \mathrm{e}^{-\gamma \cdot prog} \cdot (1 - prog)$,

  γ = decay rate

- Piecewise Linear:

  $\lambda(epoch) = \lambda_i + \frac{\lambda_{i+1} - \lambda_i}{p_{i+1} - p_i}(prog - p_i)$

  Defined by control points :$\{(p_i, \lambda_i)\}$, $p_i, \lambda_i \in [0,1]$.

  Linearly interpolated over epochs

- $epoch \in [1, epoch_{max}]$
- $prog = \frac{epoch}{epoch_{Max}} \in [0,1]$

Summary of proposed Strategies

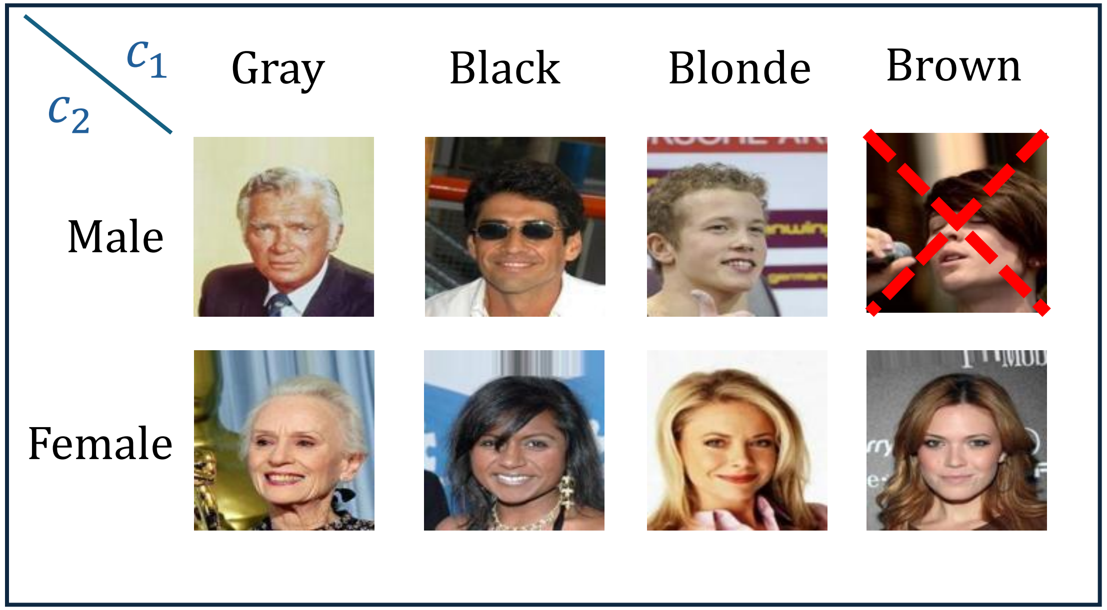
Method	Source of x_{t-k}	Description
Fully Teacher	Teacher Model Prediction (via ODE Solver)	Uses noise predicted by teacher model to estimate x_{t-k} through an ODE solver.
Fully diffusion formula	Reused Forward Process Noise	x_{t-k} formulated using the same noise as x_t.
Step Fusion	Mixed (Teacher & Diffusion Formula)	Pixel-wise weighted combination of both teacher and formulated x_{t-k}.
Loss Fusion	Both (Teacher & Diffusion Formula)	Computes separate losses, combines with weighting.
Switch	Alternating	Uses teacher before an epoch threshold, switches to formulated x_{t-k} after.

4

Experiments

Experiment setup

- Dataset preparation
  - c_1 : Hair color, 4 classes
  - c_2 : Gender, 2 classes
  - Total of 4*2=8 composition classes,
  - Unseen: (Brown hair, Male)
  - Dataset image counts: 1k~20k
  - Image size: 128*128



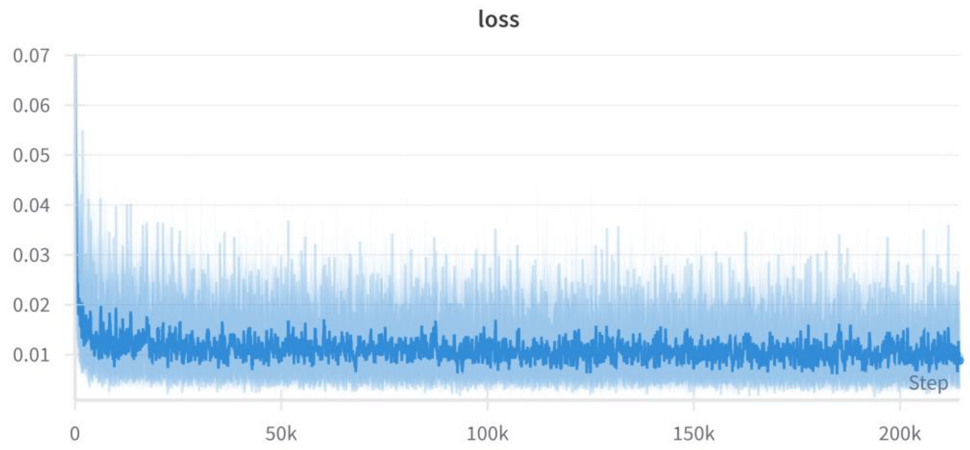
Training set with compositional class labels c_1, c_2

Experiment setup

- CCCM and Baseline Training Configuration

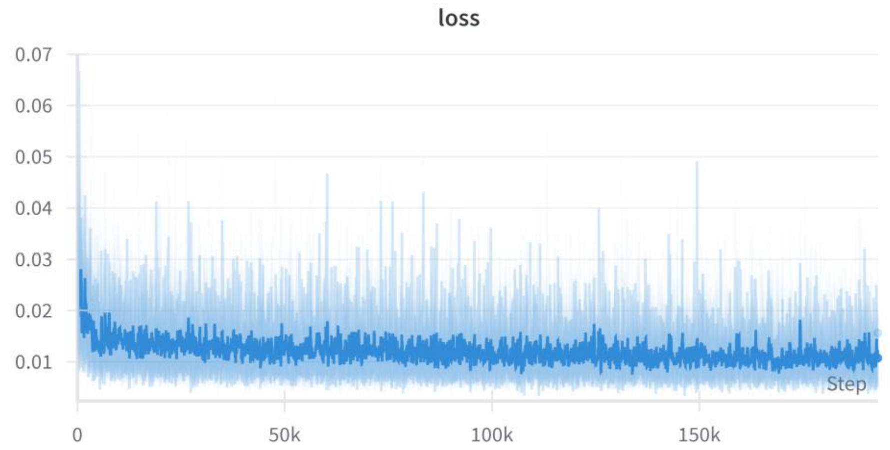
Baseline CCDM

Training Epochs	120
Learning rate, scheduler	5e-5 linear with warmup
Loss type	L2-norm



Our CCCM

Training Epochs	80
Learning rate, scheduler	5e-6 constant with warmup
Loss type	Huber loss
Guidance scale interval	2.6, 3.0
Fuse Method	Fully teacher



Experiment setup

- Modified Consistency Distillation Configuration

Fully Teacher

Fuse Method	Fully Teacher
Fuse Scheduler	Constant = 1

Fully Formulated x_{t-k}

Fuse Method	Fully Formulated
Fuse Scheduler	Constant = 0

Switch 32

Fuse Method	Switch
Fuse Scheduler	Threshold = 32

Switch 48

Fuse Method	Switch
Fuse Scheduler	Threshold = 48

Unchanged Hyperparameters

Training Epochs	80
Learning rate, scheduler	5e-6 constant with warmup
Loss type	Huber loss
Guidance scale interval	2.6, 3.0

Experiment setup

- Modified Consistency Distillation Configuration

Loss Fuse Constant

Fuse Method	Loss Fuse
Fuse Scheduler	Constant = 0.8

Loss Fuse piecewise

Fuse Method	Loss Fuse
Fuse Scheduler	Piecewise Linear, (40, 0.5)

$\lambda(epoch)$ drops to 0.5 at epoch 40 linearly, holds at 0.5 till finished.

Loss Fuse exponential

Fuse Method	Loss Fuse
Fuse Scheduler	Exponential Decay, $\gamma = 2.0$

Step Fuse exponential

Fuse Method	Step Fuse
Fuse Scheduler	Exponential Decay, $\gamma = 2.0$

Unchanged Hyperparameters

Training Epochs	80
Learning rate, scheduler	5e-6 constant with warmup
Loss type	Huber loss
Guidance scale interval	2.6, 3.0

Experiment setup

- Evaluation metrics

  - **FID** score:

    measures the similarity between generated and real image distributions.

  - Each model generates ~7k images (500 or 1000 per class) for evaluation

  - **Unseen Class Accuracy** by human evaluation:

    Compositional class $(Brown\_hair, Male)$ is generated, to test the compositional zero-shot image generation ability.

  - Each model generates 300 $(Brown\_hair, Male)$ images.

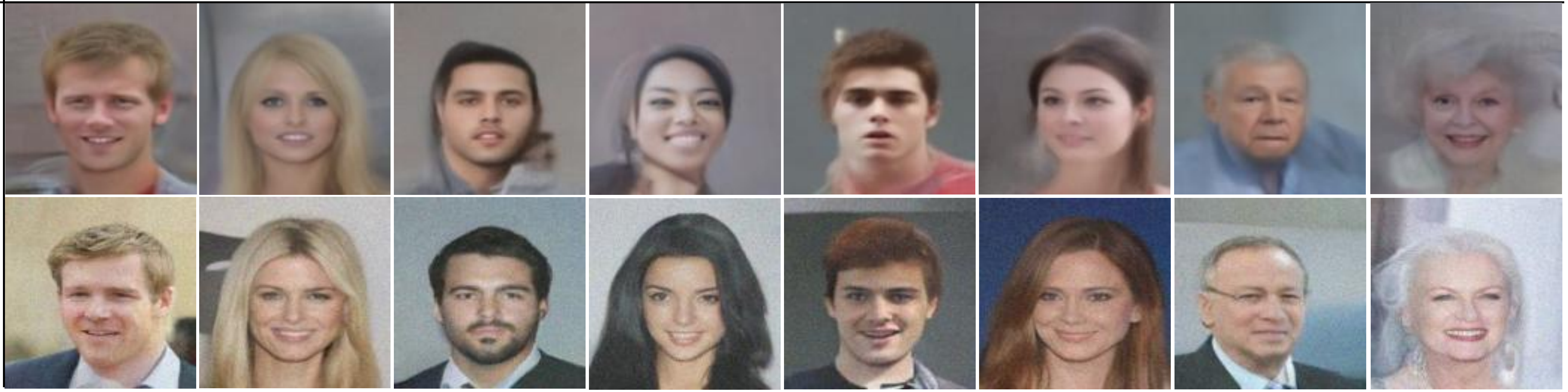
Standards:

- **Unseen class judged by both attributes.**
  Masculine features required (e.g., short hair, no makeup).
  Feminine traits → failure, even if hair color is correct.

Qualitative result

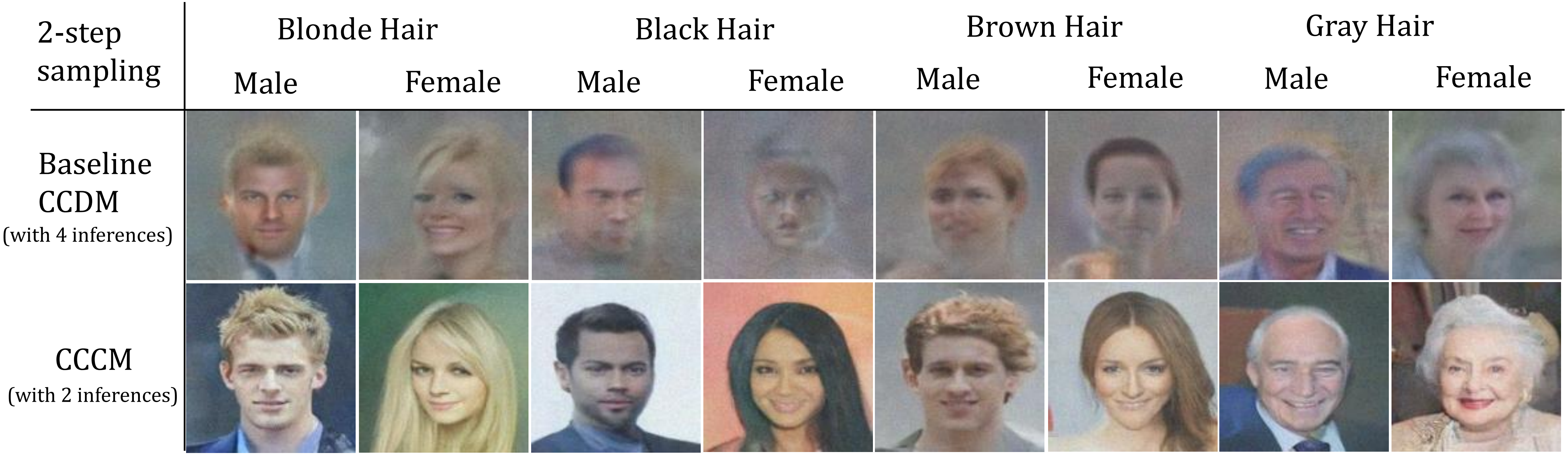
- Baseline CCDM vs Fully teacher CCCM

4-step sampling	Blonde Hair		Black Hair		Brown Hair		Gray Hair	
	Male	Female	Male	Female	Male	Female	Male	Female
Baseline CCDM (with 8 inferences)								
CCCM (with 4 inferences)								



Qualitative result

- Baseline CCDM vs Fully teacher CCCM



Quantitative result

- FID scores under 2,3 and 4 steps sampling

2 steps

Method	**FID score ↓**
Baseline DDIM	207.99
Forward-process x_{t-k}	115.66
Step Fuse (exponential)	97.13
Switch (threshold = 32)	94.99
Switch (threshold = 48)	93.97
Loss Fuse (exponential)	91.27
Loss Fuse (piecewise = 40:0.5)	88.82
Loss Fuse (constant = 0.8)	88.47
Fully Teacher x_{t-k}	**81.30**

3 steps

Method	**FID score ↓**
Baseline DDIM	136.68
Forward-process x_{t-1}	92.15
Step Fuse (exponential)	83.73
Switch (threshold = 32)	85.66
Switch (threshold = 48)	84.23
Loss Fuse (exponential)	82.90
Loss Fuse (piecewise = 40:0.5)	80.16
Loss Fuse (constant = 0.8)	77.85
Fully Teacher x_{t-k}	**73.27**

4 steps

Method	**FID score ↓**
Baseline DDIM	94.67
Diffusion-formulated x_{t-1}	80.94
Step Fuse (exponential)	77.27
Switch (threshold = 32)	76.26
Switch (threshold = 48)	75.86
Loss Fuse (exponential)	75.34
Loss Fuse (piecewise = 40:0.5)	73.45
Loss Fuse (constant = 0.8)	70.15
Fully Teacher x_{t-1}	**68.11**

Unseen Accuracy evaluation

- 300 images of Brown hair Male, compositional zero shot generation

4 steps sampling

Method	Acc% ↑
Baseline DDIM	40.6%
Diffusion-formulated $x_{t_{n-k}}$	43.0%
Step Fuse (exponential)	51.6%
Switch (threshold = 32)	49.3%
Switch (threshold = 48)	49.6%
Loss Fuse (exponential)	51.6%
Loss Fuse (piecewise = (40, 0.5))	**52.0%**
Loss Fuse (constant = 0.8)	50.6%
Fully Teacher $x_{t_{n-k}}$	47.0%

Loss Fuse
piecewise (40,05)

Fully teacher $x_{t_{n-k}}$

5.

Conclusions

Conclusion

- We propose the **Compositional Conditional Consistency Model.**
- Achieving **faster sampling** speed than CCDM.
- **Preserves unseen** class generation.
- Observed that modified consistency distillation strategies yield **better unseen accuracy.**

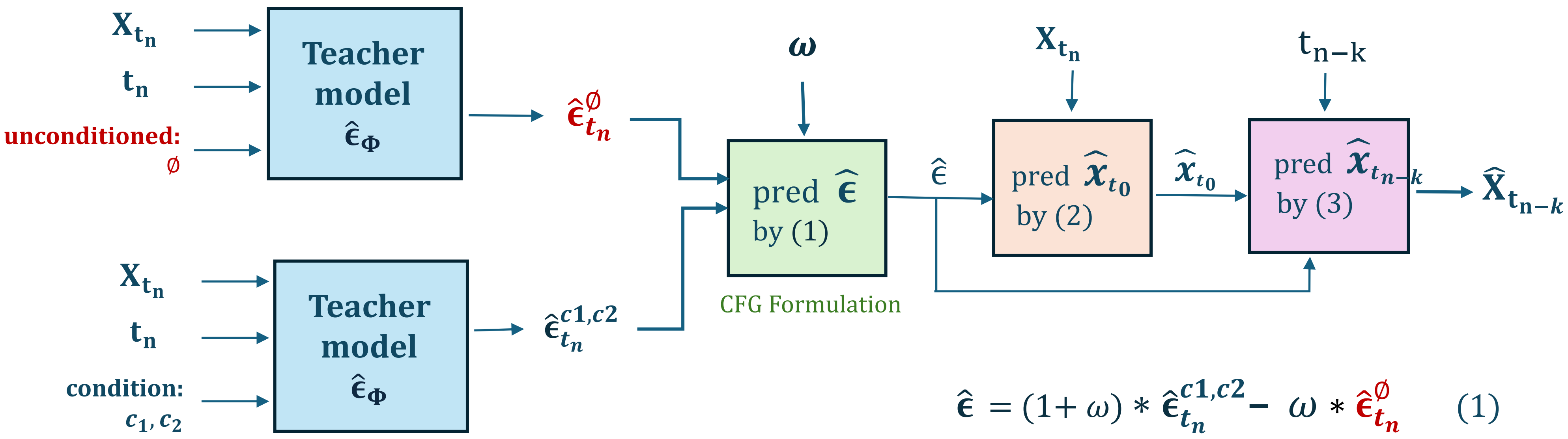
  Our guess is: the teacher model may introduce **bias**, encouraging the student to generate seen or **high-confidence images**, which **limits generalization**.

References

- **"Compositional Conditional Diffusion Model",** S. -L. Lai, P. -C. Chen & C. -W. Ma, 2024.
- "**Consistency Models",** Y. Song, P. Dhariwal, M. Chen & I. Sutskever, 2023.
- **"Latent consistency models: Synthesizing high-resolution images with few-step inference"**, Luo, S., Tan, Y., Huang, L., Li, J., & Zhao, H. 2023.
- **"Improved techniques for training consistency models",** Y. Song and P. Dhariwal, 2023.
- **"Score-based generative modeling through stochastic differential equations",** Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon & B. Poole, 2020.
- **"Denoising diffusion implicit models",** J. Song, C. Meng & S. Ermon**,** 2020.
- **"Progressive distillation for fast sampling of diffusion models",** T. Salimans and J. Ho, 2022.
- **"On distillation of guided diffusion models",** C. Meng, R. Rombach, R. Gao, D. Kingma, S. Ermon, J. Ho, and T. Salimans, 2023.
- **"Classifier-free diffusion guidance",** J. Ho and T. Salimans, 2022.
- **"High-resolution image synthesis with latent diffusion models",** R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, 2022.

ODE solver

- CCCM uses DDIM as ODE solver



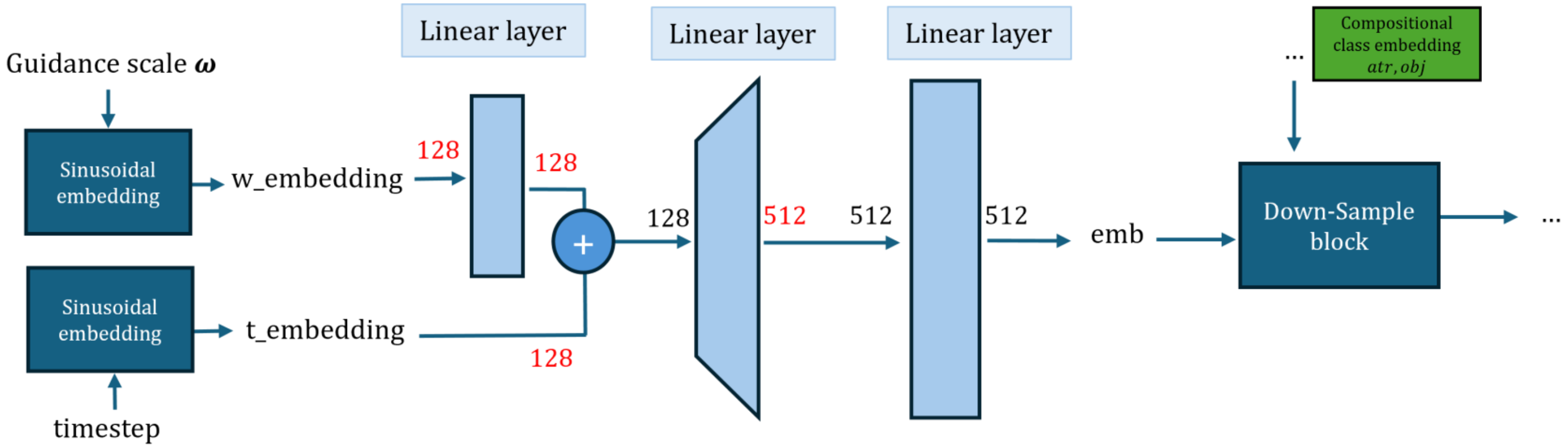
$$\hat{\epsilon} = (1+\omega) * \hat{\epsilon}_{t_n}^{c1,c2} - \omega * \hat{\epsilon}_{t_n}^{\emptyset} \quad (1)$$

$$\hat{X}_{t_0} = \frac{X_{t_n} - \sigma_{t_n} * \hat{\epsilon}}{\alpha_{t_n}} \quad (2)$$

$$\hat{X}_{t_{n-k}} = \alpha_{t_{n-k}} * \hat{X}_0 + \sigma_{t_{n-k}} * \hat{\epsilon} \quad (3)$$

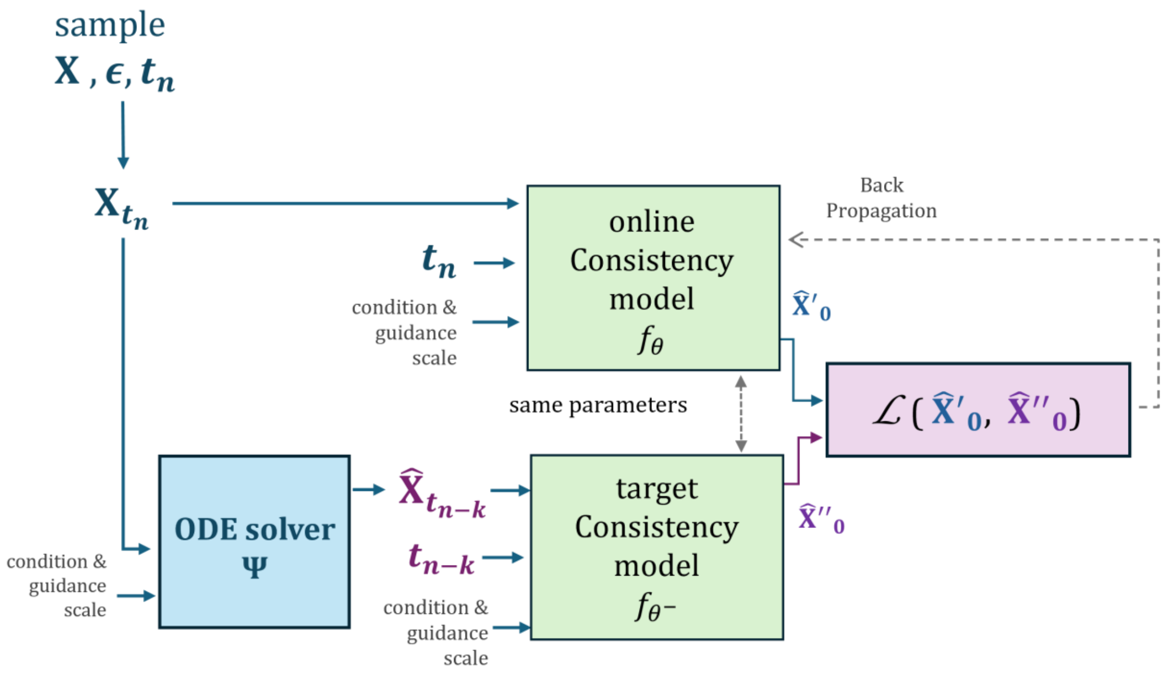
Notations:
t_n : time at step n, $t_n \in [0,1], n \in [1..T]$
α_{t_n} : signal rate ; σ_{t_n} : noise rate
X_{t_n}: image at t_n ; $\hat{X}_{t_n}$: predicted image at t_n
$\hat{\epsilon}_{t_n}$: predicted noise at t_n
ω : guidance scale

Guidance scale embedding



Compositional Consistency Distillation Algo

- Fully teacher



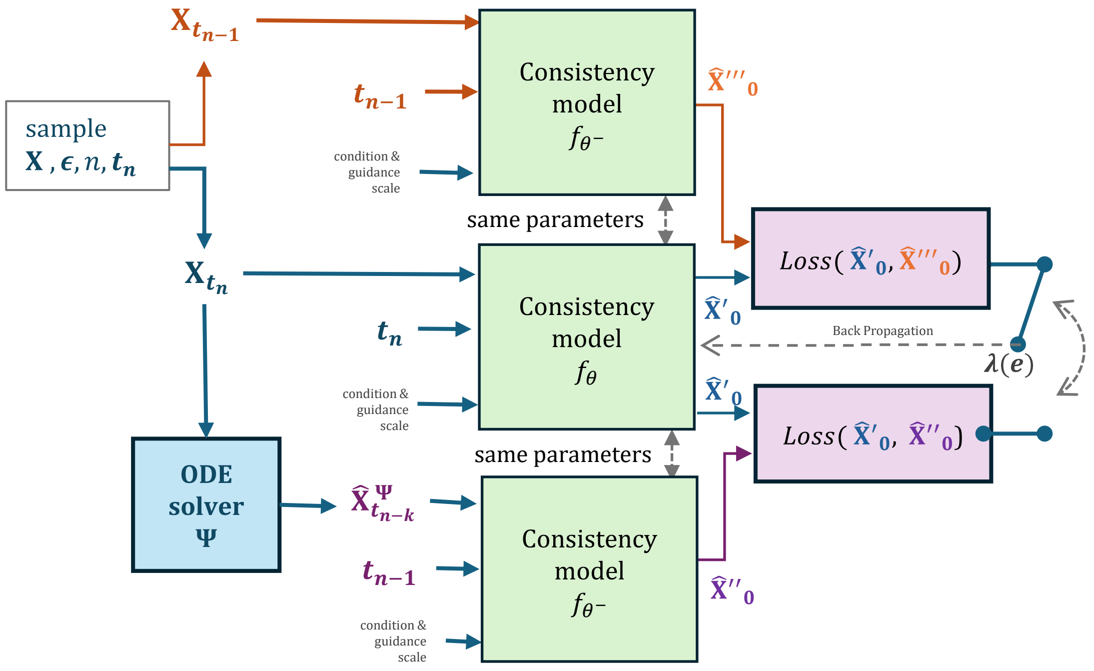
Algorithm 4.1 Compositional label Consistency Distillation

Input:dataset $\mathcal{D}$, initial model parameter θ, learning rate η, ODE solver $\Psi(\cdot,\cdot,\cdot,\cdot)$, distance metric $d(\cdot,\cdot)$, guidance scale $[\omega_{min},\omega_{max}]$, skipping steps k

1: $\theta^- \leftarrow \theta$
2: **repeat**
3: Sample $(x, c_1, c_2) \sim \mathcal{D}$, and $\omega \sim [\omega_{min},\omega_{max}]$
4: Sample $\epsilon \sim \mathcal{N}(0, I),\ n \sim \mathcal{U}[1+k, N]$
5: $x_{t_n} \leftarrow \sqrt{\bar\alpha_{t_n}} \cdot x + \sqrt{1-\bar\alpha_{t_n}} \cdot \epsilon$
6: $x^{\Psi,\omega}_{t_{n-k}} \leftarrow x_{t_n} + (1+\omega)\cdot\Psi(x_{t_n}, t_n, t_{n-k}, c_1, c_2) - \omega\cdot\Psi(x_{t_n}, t_n, t_{n-k}, \varnothing, \varnothing)$
7: $\mathcal{L}(\theta,\theta^-;\Psi) \leftarrow d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega),\ f_{\theta^-}(\hat{x}_{t_{n-k}}, t_{n-k}, c_1, c_2, \omega)\right)$
8: $\theta \leftarrow \theta - \eta\nabla_\theta\mathcal{L}(\theta,\theta^-)$
9: $\theta^- \leftarrow \texttt{stopgrad}\,(\mu\theta^- + (1-\mu)\,\theta)$
10: **until** convergence

Compositional Consistency Distillation Algo

- Switch



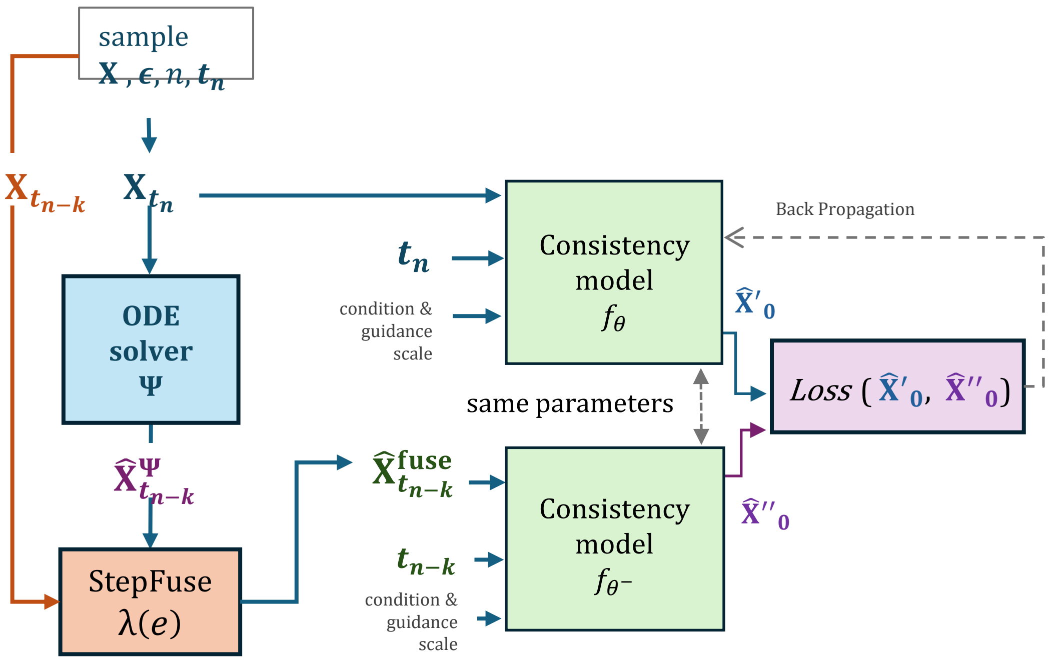
Algorithm 4.2 Modified Consistency Distillation–Switch

Input:dataset $\mathcal{D}$, initial model parameter θ, learning rate η, ODE solver $\Psi(\cdot,\cdot,\cdot,\cdot)$, distance metric $d(\cdot,\cdot)$, guidance scale $[\omega_{min}, \omega_{max}]$, skipping steps k, **switching threshold** e_{switch}

$\theta^- \leftarrow \theta$
repeat
  Sample $(x, c_1, c_2) \sim \mathcal{D}$, $\omega \sim [\omega_{\min}, \omega_{\max}]$
  Sample $\epsilon \sim \mathcal{N}(0, I)$, $n \sim \mathcal{U}[1+k, N]$
  $x_{t_n} \leftarrow \sqrt{\bar{\alpha}_{t_n}} \cdot x + \sqrt{1-\bar{\alpha}_{t_n}} \cdot \epsilon$
  if $e < e_{\text{switch}}$ **then**
    $x^{\Psi,\omega}_{t_{n-k}} \leftarrow x_{t_n} + (1+\omega) \cdot \Psi(x_{t_n}, t_n, t_{n-k}, c_1, c_2) - \omega \cdot \Psi(x_{t_n}, t_n, t_{n-k}, \varnothing, \varnothing)$
    $\mathcal{L}_{\text{Switch}}(\theta, \theta^-; \Psi) \leftarrow d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega),\ f_{\theta^-}(x^{\Psi,\omega}_{t_{n-k}}, t_{n-k}, c_1, c_2, \omega)\right)$
  else
    $x_{t_{n-k}} \leftarrow \sqrt{\bar{\alpha}_{t_{n-k}}} \cdot x + \sqrt{1-\bar{\alpha}_{t_{n-k}}} \cdot \epsilon$
    $\mathcal{L}_{\text{Switch}}(\theta, \theta^-) \leftarrow d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega),\ f_{\theta^-}(x_{t_{n-k}}, t_{n-k}, c_1, c_2, \omega)\right)$
  end if
  $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{\text{Switch}}(\theta, \theta^-)$
  $\theta^- \leftarrow \texttt{stopgrad}\left(\mu\theta^- + (1-\mu)\theta\right)$
until convergence

Compositional Consistency Distillation Algo

- StepFuse



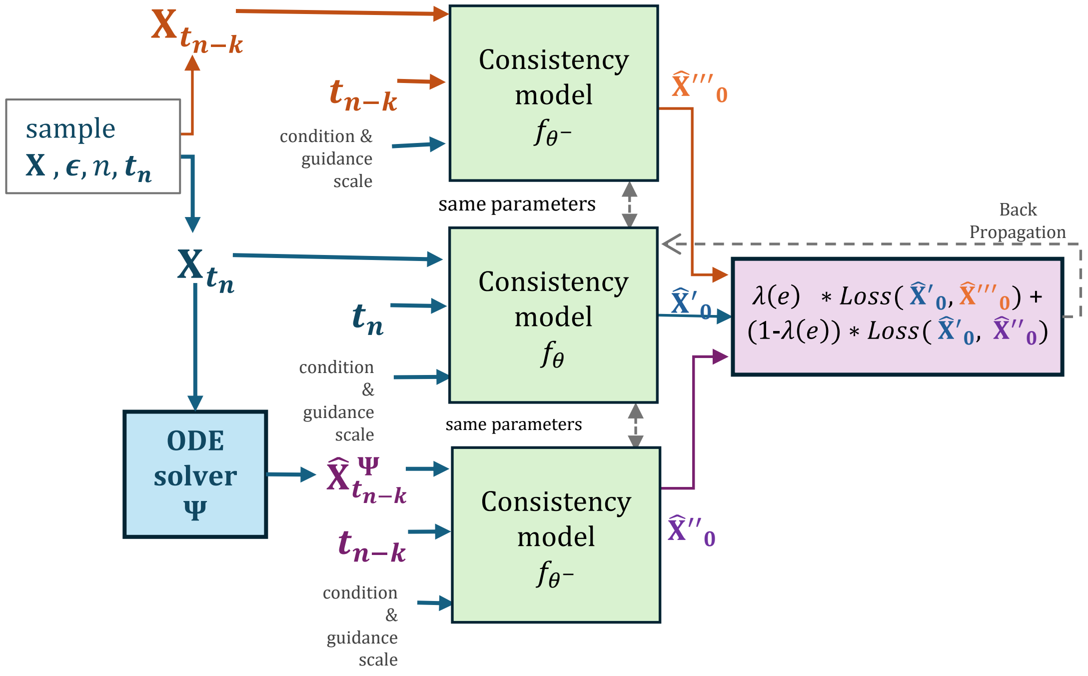
Algorithm 4.3 Modified Compositional label Consistency Distillation–StepFuse

Input:dataset $\mathcal{D}$, initial model parameter θ, learning rate η, ODE solver $\Psi(\cdot,\cdot,\cdot,\cdot)$, distance metric $d(\cdot,\cdot)$, guidance scale $[\omega_{min}, \omega_{max}]$, skipping steps k, **fuse scheduler** $\lambda(e)$

$\theta^- \leftarrow \theta$
repeat
  Sample $(x, c_1, c_2) \sim \mathcal{D}$, and $\omega \sim [\omega_{min}, \omega_{max}]$
  Sample $\epsilon \sim \mathcal{N}(0, I)$, $n \sim \mathcal{U}[1+k, N]$
  $x_{t_n} \leftarrow \sqrt{\bar{\alpha}_{t_n}} \cdot x + \sqrt{1-\bar{\alpha}_{t_n}} \cdot \epsilon$
  $x^{\Psi,\omega}_{t_{n-k}} \leftarrow x_{t_n} + (1+\omega) \cdot \Psi(x_{t_n}, t_n, t_{n-k}, c_1, c_2) - \omega \cdot \Psi(x_{t_n}, t_n, t_{n-k}, \varnothing, \varnothing)$
  $x_{t_{n-k}} \leftarrow \sqrt{\bar{\alpha}_{t_{n-k}}} \cdot x + \sqrt{1-\bar{\alpha}_{t_{n-k}}} \cdot \epsilon$
  $\hat{x}^{\text{fuse}}_{t-k} \leftarrow \lambda(e) \cdot \hat{x}^{\Psi,\omega}_{t_{n-k}} + (1-\lambda(e)) \cdot x_{t-k}$
  $\mathcal{L}_{Stepfuse}(\theta, \theta^-; \Psi) \leftarrow d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega),\ f_{\theta^-}(\hat{x}^{\text{fuse}}_{t-k}, t_{n-k}, c_1, c_2, \omega)\right)$
  $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{Stepfuse}(\theta, \theta^-)$
  $\theta^- \leftarrow \texttt{stopgrad}\,(\mu\theta^- + (1-\mu)\,\theta)$
until convergence

Compositional Consistency Distillation Algo

- LossFuse



Algorithm 4.4 Modified Compositional label Consistency Distillation–LossFuse

Input:dataset $\mathcal{D}$, initial model parameter θ, learning rate η, ODE solver $\Psi(\cdot,\cdot,\cdot,\cdot)$, distance metric $d(\cdot,\cdot)$, guidance scale $[\omega_{min}, \omega_{max}]$, skipping steps k, **fuse scheduler** $\lambda(e)$

$\theta^- \leftarrow \theta$
repeat
  Sample $(x, c_1, c_2) \sim \mathcal{D}$, and $\omega \sim [\omega_{min}, \omega_{max}]$
  Sample $\epsilon \sim \mathcal{N}(0, I)$, $n \sim \mathcal{U}[1+k, N]$
  $x_{t_n} \leftarrow \sqrt{\bar{\alpha}_{t_n}} \cdot x + \sqrt{1-\bar{\alpha}_{t_n}} \cdot \epsilon$
  $x^{\Psi,\omega}_{t_{n-k}} \leftarrow x_{t_n} + (1+\omega) \cdot \Psi(x_{t_n}, t_n, t_{n-k}, c_1, c_2) - \omega \cdot \Psi(x_{t_n}, t_n, t_{n-k}, \varnothing, \varnothing)$
  $x_{t_{n-k}} \leftarrow \sqrt{\bar{\alpha}_{t_{n-k}}} \cdot x + \sqrt{1-\bar{\alpha}_{t_{n-k}}} \cdot \epsilon$
  $\mathcal{L}_{teacher}(\theta, \theta^-; \Psi) \leftarrow d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega),\ f_{\theta^-}(\hat{x}^{\Psi,\omega}_{t-k}, t_{n-k}, c_1, c_2, \omega)\right)$
  $\mathcal{L}_{fpf}(\theta, \theta^-) \leftarrow d\left(f_\theta(x_{t_n}, t_n, c_1, c_2, \omega),\ f_{\theta^-}(x_{t-k}, t_{n-k}, c_1, c_2, \omega)\right)$
  $\mathcal{L}_{lossfuse} \leftarrow \lambda(e) \cdot \mathcal{L}_{teacher} + (1-\lambda(e)) \cdot \mathcal{L}_{fpf}$
  $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_{lossfuse}(\theta, \theta^-)$
  $\theta^- \leftarrow \texttt{stopgrad}(\mu\theta^- + (1-\mu)\theta)$
until convergence

Experiment setup

- Training Configuration

Hyperparameters fixed across all experiments

Training timesteps	1000
Batch Size	24
Noise scheduler β	0.0001-0.2 linear
Base channels dimension	128
# of Residual Blocks	2
Channel Multiplier	1,2,4,4
Optimizer	AdamW
ω embedding dimension	128
Huber loss param	$\delta = 0.001$
EMA decay	$\mu = 0.995$