Flow Matching for Generative Modeling

Dorian Gailhard

doriangailhard.github.io

June 25, 2026

Context: Generative Modeling

We want to model the distribution of a dataset of images:

$$x \sim p_{\text{data}}(x)$$

Context: Generative Modeling

We want to model the distribution of a dataset of images:

$$x \sim p_{\text{data}}(x)$$

Direct sampling from p_{data} is difficult.

Context: Generative Modeling

We want to model the distribution of a dataset of images:

$$x \sim p_{\text{data}}(x)$$

Direct sampling from p_{data} is difficult.
Idea: learn a transformation f such that

$$x = f(z), \quad z \sim p_0 \text{ where } p_0 \text{ is easy to sample}$$

Context: Generative Modeling

We want to model the distribution of a dataset of images:

$$x \sim p_{\text{data}}(x)$$

Direct sampling from p_{data} is difficult.
Idea: learn a transformation f such that

$$x = f(z), \quad z \sim p_0 \text{ where } p_0 \text{ is easy to sample}$$

This is known as a **transport map** (or **pushforward**):

$$p_{\text{data}} = f_{\#} p_z$$

Context: Generative Modeling

We want to model the distribution of a dataset of images:

$$x \sim p_{\text{data}}(x)$$

Direct sampling from p_{data} is difficult.
Idea: learn a transformation f such that

$$x = f(z), \quad z \sim p_0 \text{ where } p_0 \text{ is easy to sample}$$

This is known as a **transport map** (or **pushforward**):

$$p_{\text{data}} = f_{\#} p_z$$

Since sampling p_0 is easy, the only difficulty is learning f.

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Instead of learning f directly, define a **continuous flow**:

$$\frac{dx_t}{dt} = v_t^*(x_t),$$

such that: $x_1 = x_0 + \int_0^1 v_t^*(x_t)\, dt$, where $x_0 \sim p_0$, $x_1 \sim p_{\text{data}}$.

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Instead of learning f directly, define a **continuous flow**:

$$\frac{dx_t}{dt} = v_t^*(x_t),$$

such that: $x_1 = x_0 + \int_0^1 v_t^*(x_t)\, dt$, where $x_0 \sim p_0$, $x_1 \sim p_{\text{data}}$.

We can learn v_t^* using a neural network.

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Instead of learning f directly, define a **continuous flow**:

$$\frac{dx_t}{dt} = v_t^*(x_t),$$

such that: $x_1 = x_0 + \int_0^1 v_t^*(x_t)\, dt$, where $x_0 \sim p_0$, $x_1 \sim p_{\text{data}}$.

We can learn v_t^* using a neural network.

Flow Matching idea:

- Define a target probability path (x_t)
- Learn v_t to match the displacement along this path

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Instead of learning f directly, define a **continuous flow**:

$$\frac{dx_t}{dt} = v_t^*(x_t),$$

such that: $x_1 = x_0 + \int_0^1 v_t^*(x_t)\,dt$, where $x_0 \sim p_0$, $x_1 \sim p_{\text{data}}$.

We can learn v_t^* using a neural network.

Flow Matching idea:

- Define a target probability path (x_t)
- Learn v_t to match the displacement along this path

$$x_1 = x_0 + \int_0^1 v_t^*(x_t)\,dt$$

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Instead of learning f directly, define a **continuous flow**:

$$\frac{dx_t}{dt} = v_t^*(x_t),$$

such that: $x_1 = x_0 + \int_0^1 v_t^*(x_t)\,dt$, where $x_0 \sim p_0$, $x_1 \sim p_{\text{data}}$.

We can learn v_t^* using a neural network.

Flow Matching idea:

- Define a target probability path (x_t)
- Learn v_t to match the displacement along this path

$$x_1 = x_0 + \int_0^1 v_t^*(x_t)\,dt \approx x_0 + \int_0^1 v_t^\theta(x_t)\,dt$$

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Starting Distribution p_0:

- Usually chosen as a simple distribution (e.g., Gaussian) for easy sampling: $p_0 = \mathcal{N}(0, I)$

Starting Distribution p_0:

- Usually chosen as a simple distribution (e.g., Gaussian) for easy sampling: $p_0 = \mathcal{N}(0, I)$
- In principle, any tractable distribution can be used

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Starting Distribution p_0:

- Usually chosen as a simple distribution (e.g., Gaussian) for easy sampling: $p_0 = \mathcal{N}(0, I)$
- In principle, any tractable distribution can be used

Probability Paths $(x_t)_{t\in[0,1]}$:

- There are infinitely many paths connecting p_0 to p_1

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Starting Distribution p_0:

- Usually chosen as a simple distribution (e.g., Gaussian) for easy sampling: $p_0 = \mathcal{N}(0, I)$
- In principle, any tractable distribution can be used

Probability Paths $(x_t)_{t\in[0,1]}$:

- There are infinitely many paths connecting p_0 to p_1
- Most commonly used: **straight paths** $x_t = (1-t)x_0 + tx_1$

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Starting Distribution p_0:

- Usually chosen as a simple distribution (e.g., Gaussian) for easy sampling: $p_0 = \mathcal{N}(0, I)$
- In principle, any tractable distribution can be used

Probability Paths $(x_t)_{t\in[0,1]}$:

- There are infinitely many paths connecting p_0 to p_1
- Most commonly used: **straight paths** $x_t = (1-t)x_0 + tx_1$
- Other paths can be used depending on the application

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Starting Distribution p_0:

- Usually chosen as a simple distribution (e.g., Gaussian) for easy sampling: $p_0 = \mathcal{N}(0, I)$
- In principle, any tractable distribution can be used

Probability Paths $(x_t)_{t\in[0,1]}$:

- There are infinitely many paths connecting p_0 to p_1
- Most commonly used: **straight paths** $x_t = (1-t)x_0 + tx_1$
- Other paths can be used depending on the application

Model Prediction / Loss: The network predicts a velocity field $v_t^\theta(x_t)$ along the chosen path, trained with:

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0,x_1,t}\left[\left\| v_t^\theta(x_t) - \dot{x}_t \right\|^2\right]$$

where $\dot{x}_t$ is the derivative of the chosen path at time t.

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Examples of Probability Paths:

- **Straight / linear paths:** $x_t = (1-t)x_0 + tx_1$ — simplest deterministic interpolation

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Examples of Probability Paths:

- **Straight / linear paths:** $x_t = (1-t)x_0 + tx_1$ — simplest deterministic interpolation
- **Diffusion-style Variance-Exploding path:** Conditional vector field:

$$w_t(x \mid x_0) = -\frac{\sigma'_{1-t}}{\sigma_{1-t}}(x - x_0)$$

where σ_t is the noise schedule. Equivalent to VP or VE schedule in diffusion models.

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Examples of Probability Paths:

- **Straight / linear paths:** $x_t = (1-t)x_0 + tx_1$ — simplest deterministic interpolation
- **Diffusion-style Variance-Exploding path:** Conditional vector field:

$$w_t(x \mid x_0) = -\frac{\sigma'_{1-t}}{\sigma_{1-t}}(x - x_0)$$

  where σ_t is the noise schedule. Equivalent to VP or VE schedule in diffusion models.
- **Dirichlet flows (on simplex):** Conditional paths: $p_t(x \mid x_1 = e_i) = \mathrm{Dir}(x; \alpha = 1 + t \cdot e_i)$, initial $p_0 = \mathrm{Dir}(x; \alpha = (1, \ldots, 1)^T)$

  Ref: *Dirichlet Flow Matching with Applications to DNA Sequence Design*, Hannes Stark et al.

Flow Matching vs. DDIM

DDIM:

- To generate a sample from noise to data:

$$x_1 = x_0 + \int_0^1 v_t \, dt, \quad v_t = \frac{f^\theta(x_t, t) - x_t}{1 - t}$$

Flow Matching vs. DDIM

DDIM:

- To generate a sample from noise to data:

$$x_1 = x_0 + \int_0^1 v_t \, dt, \quad v_t = \frac{f^\theta(x_t, t) - x_t}{1 - t}$$

- Learned by minimizing the conditional prediction loss along the path:

$$\mathcal{L}_{\text{DDIM}} = \mathbb{E}_{x_0, x_1, t}\left[\|f^\theta(x_t, t) - x_1\|^2\right]$$

Flow Matching vs. DDIM

DDIM:

- To generate a sample from noise to data:

$$x_1 = x_0 + \int_0^1 v_t \, dt, \quad v_t = \frac{f^\theta(x_t, t) - x_t}{1 - t}$$

- Learned by minimizing the conditional prediction loss along the path:

$$\mathcal{L}_{\text{DDIM}} = \mathbb{E}_{x_0, x_1, t}\left[\|f^\theta(x_t, t) - x_1\|^2\right]$$

FM:

- To generate a sample from noise to data:

$$x_1 = x_0 + \int_0^1 v_t^\theta \, dt$$

Flow Matching vs. DDIM

DDIM:

- To generate a sample from noise to data:

$$x_1 = x_0 + \int_0^1 v_t \, dt, \quad v_t = \frac{f^\theta(x_t, t) - x_t}{1 - t}$$

- Learned by minimizing the conditional prediction loss along the path:

$$\mathcal{L}_{\text{DDIM}} = \mathbb{E}_{x_0, x_1, t}\left[\|f^\theta(x_t, t) - x_1\|^2\right]$$

FM:

- To generate a sample from noise to data:

$$x_1 = x_0 + \int_0^1 v_t^\theta \, dt$$

- Learned by minimizing the conditional prediction loss along the path:

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{x_0, x_1, t}\left[\|v^\theta(x_t, t) - (x_1 - x_0)\|^2\right]$$

Flow Matching for Generative Modeling

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, Matt Le

Equivalent Formulations and Concurrent Work:

- **Stochastic Interpolants: A Unifying Framework for Flows and Diffusions**

  Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden

- **Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow**

  Xingchao Liu, Chengyue Gong, Qiang Liu

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

- Standard Flow Matching samples $x_0 \sim p_0$ and $x_1 \sim p_1$ **independently**.

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

- Standard Flow Matching samples $x_0 \sim p_0$ and $x_1 \sim p_1$ **independently**.
- This independence leads to targets with **huge variance**, slowing learning.

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

- Standard Flow Matching samples $x_0 \sim p_0$ and $x_1 \sim p_1$ **independently**.
- This independence leads to targets with **huge variance**, slowing learning.
- **Idea:** Correlate x_0 and x_1 across the minibatch to produce straighter paths and more efficient learning.

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

Multisample Flow Matching Algorithm:

1. Sample minibatches: $\{x_0^{(i)}\}_{i=1}^k \sim q_0(x_0)$, $\{x_1^{(i)}\}_{i=1}^k \sim q_1(x_1)$

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

Multisample Flow Matching Algorithm:

1. Sample minibatches: $\{x_0^{(i)}\}_{i=1}^k \sim q_0(x_0)$, $\{x_1^{(i)}\}_{i=1}^k \sim q_1(x_1)$
2. Construct a **doubly-stochastic matrix** $\pi(i,j)$ dependent on the samples

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

Multisample Flow Matching Algorithm:

1. Sample minibatches: $\{x_0^{(i)}\}_{i=1}^k \sim q_0(x_0)$, $\{x_1^{(i)}\}_{i=1}^k \sim q_1(x_1)$
2. Construct a **doubly-stochastic matrix** $\pi(i,j)$ dependent on the samples
3. Define the joint discrete distribution:

$$q_k(x_0, x_1) = \frac{1}{k} \sum_{i,j=1}^{k} \delta(x_0 - x_0^{(i)}) \delta(x_1 - x_1^{(j)}) \pi(i,j)$$

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

Multisample Flow Matching Algorithm:

1. Sample minibatches: $\{x_0^{(i)}\}_{i=1}^k \sim q_0(x_0)$, $\{x_1^{(i)}\}_{i=1}^k \sim q_1(x_1)$
2. Construct a **doubly-stochastic matrix** $\pi(i,j)$ dependent on the samples
3. Define the joint discrete distribution:

$$q_k(x_0, x_1) = \frac{1}{k} \sum_{i,j=1}^{k} \delta(x_0 - x_0^{(i)}) \delta(x_1 - x_1^{(j)}) \pi(i,j)$$

4. Train the velocity field along interpolated paths sampled from $q_k(x_0, x_1)$

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

How to choose the minibatch coupling π:

- **Batch Optimal Transport (BatchOT):** Compute the exact OT plan between minibatch samples to produce a *permutation-like* doubly-stochastic matrix. Leads to nearly straight interpolated paths. Complexity: $O(k^3)$ for a batch of size k (Hungarian or network simplex algorithms).

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

How to choose the minibatch coupling π:

- **Batch Optimal Transport (BatchOT):** Compute the exact OT plan between minibatch samples to produce a *permutation-like* doubly-stochastic matrix. Leads to nearly straight interpolated paths. Complexity: $O(k^3)$ for a batch of size k (Hungarian or network simplex algorithms).
- **Batch Entropic OT (BatchEOT):** Introduce an entropic regularization term to approximate BatchOT efficiently. Solved via Sinkhorn iterations, runtime $\tilde{O}(k^2/\epsilon)$. Outputs a doubly-stochastic matrix close to independent coupling, cheaper for large k.

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

How to choose the minibatch coupling π:

- **Batch Optimal Transport (BatchOT):** Compute the exact OT plan between minibatch samples to produce a *permutation-like* doubly-stochastic matrix. Leads to nearly straight interpolated paths. Complexity: $O(k^3)$ for a batch of size k (Hungarian or network simplex algorithms).
- **Batch Entropic OT (BatchEOT):** Introduce an entropic regularization term to approximate BatchOT efficiently. Solved via Sinkhorn iterations, runtime $\tilde{O}(k^2/\epsilon)$. Outputs a doubly-stochastic matrix close to independent coupling, cheaper for large k.

More prosaicly, reassign (in a hard or soft manner) the starting noises to the endpoints, to minimize the total target displacement.

Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

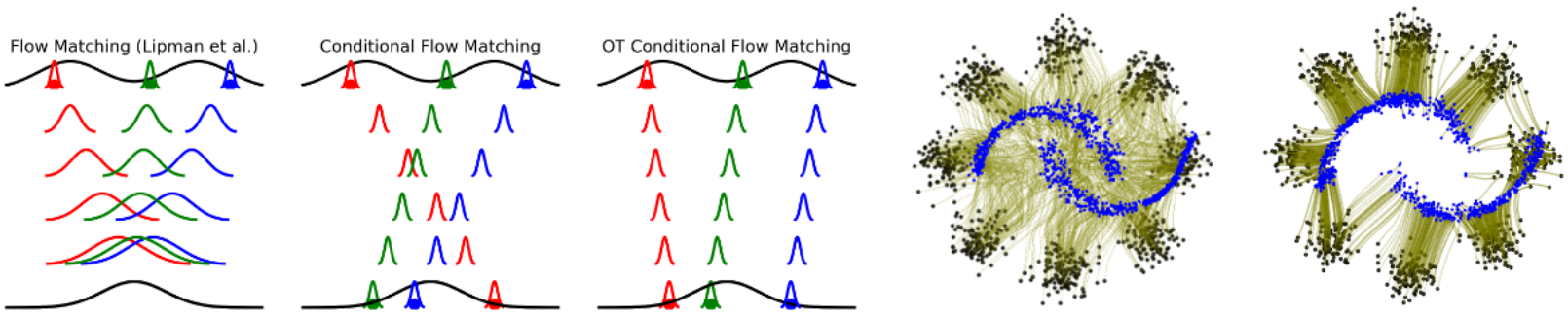
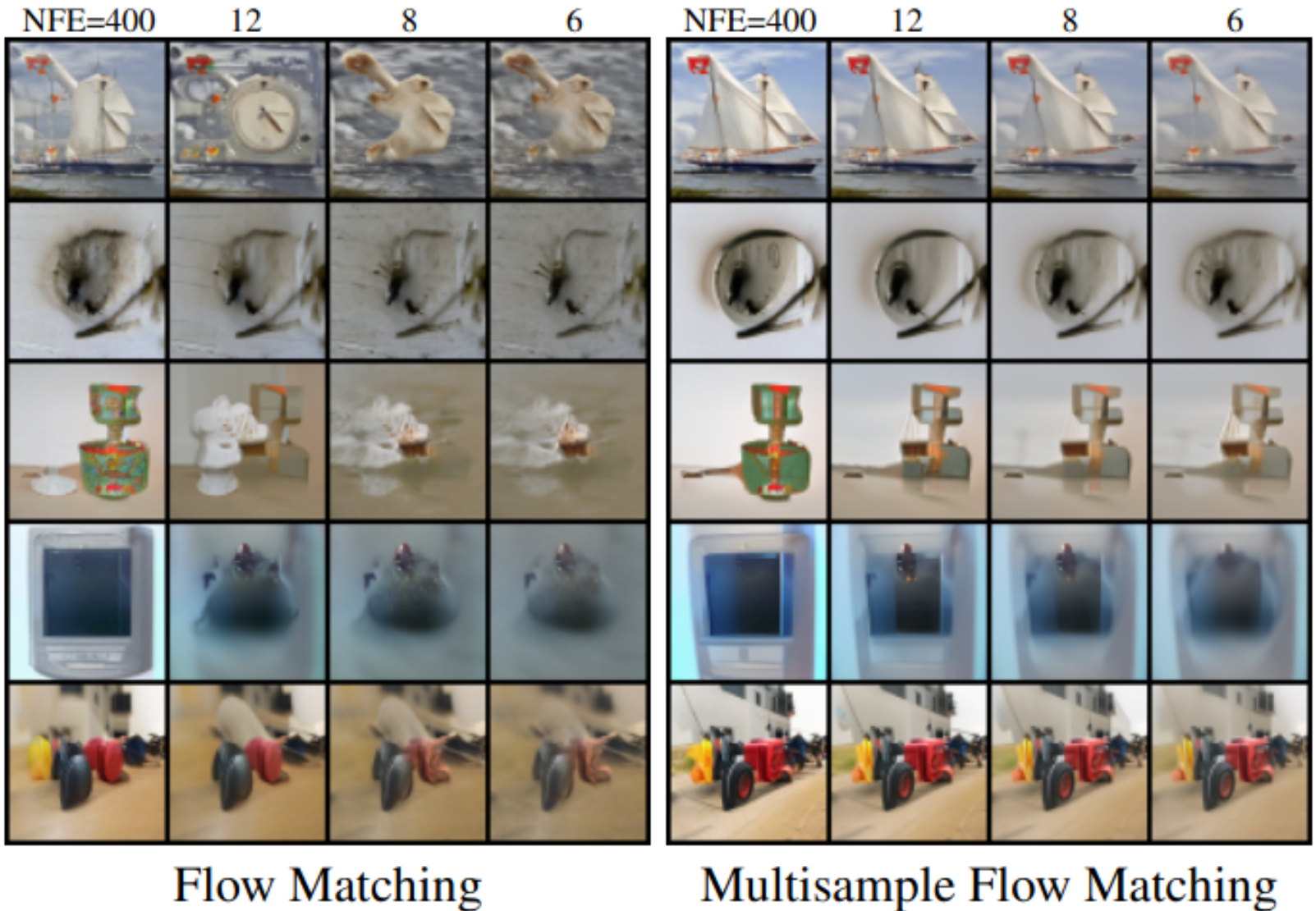


Figure: Reassigning prior samples "uncrosses" the target trajectories.

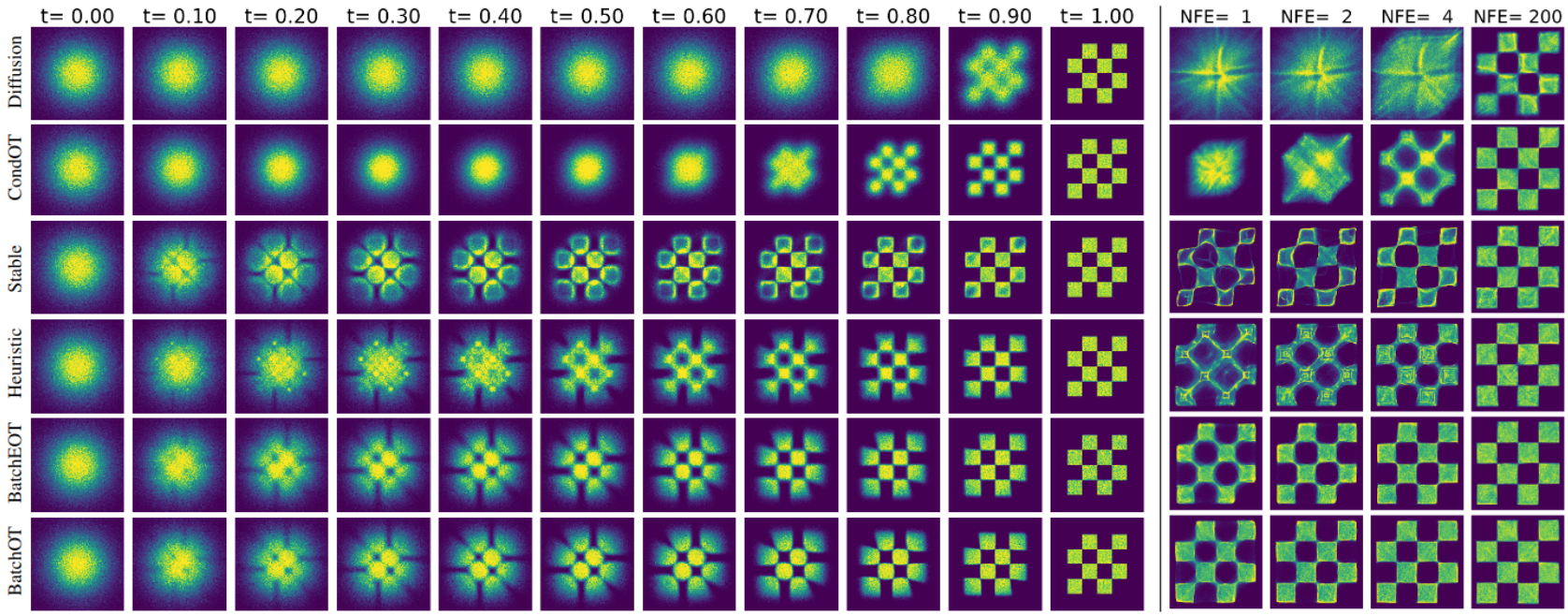
Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen



Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen



Multisample Flow Matching: Straightening Flows with Minibatch Couplings

Aram-Alexandre Pooladian, Heli Ben-Hamu, Carles Domingo-Enrich, Brandon Amos, Yaron Lipman, Ricky T. Q. Chen

Equivalent formulation and Concurrent Work:

- **Improving and Generalizing Flow-Based Generative Models with Minibatch Optimal Transport**

  Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, Yoshua Bengio

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

- Recall the Flow Matching loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0 \sim p_0, x_1 \sim p_{\text{data}}, t \sim U([0,1])} \left[\| u_\theta(x_t, t) - \dot{x}_t \|^2 \right],$$

where $x_t = (1-t)x_0 + t x_1$

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

- Recall the Flow Matching loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{x_0 \sim p_0, x_1 \sim p_{\text{data}}, t \sim U([0,1])} \left[\| u_\theta(x_t, t) - \dot{x}_t \|^2 \right],$$

where $x_t = (1-t)x_0 + t x_1$

- The optimal velocity field for this loss has a closed-form solution. No stochasticity in the target is needed.

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

Closed-form:

$$\text{If } x_t = (1-t)x_0 + t x_1, \quad b(1) := x_1$$

$$\text{then } \hat{u}^\star_M(x,t) = \sum_{j=1}^{M} \lambda(x,t)\frac{b(j)-x}{1-t},$$

$$\text{where } \lambda(x,t) = \text{softmax}\Big[-\frac{\|x - t\,b(l)\|^2}{2(1-t)^2}\Big]_{l=1,\dots,M}$$

- $b(j)$: samples from the dataset

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

Closed-form:

$$\text{If } x_t = (1-t)x_0 + t x_1, \quad b(1) := x_1$$

$$\text{then } \hat{u}^\star_M(x,t) = \sum_{j=1}^{M} \lambda(x,t)\frac{b(j)-x}{1-t},$$

$$\text{where } \lambda(x,t) = \text{softmax}\Big[-\frac{\|x - t\,b(l)\|^2}{2(1-t)^2}\Big]_{l=1,\dots,M}$$

- $b(j)$: samples from the dataset
- Becomes a "classical" ML compression problem.

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

Closed-form:

$$\text{If } x_t = (1-t)x_0 + t x_1, \quad b(1) := x_1$$

$$\text{then } \hat{u}^\star_M(x,t) = \sum_{j=1}^{M} \lambda(x,t)\frac{b(j)-x}{1-t},$$

$$\text{where } \lambda(x,t) = \text{softmax}\Big[-\frac{\|x - t\,b(l)\|^2}{2(1-t)^2}\Big]_{l=1,\dots,M}$$

- $b(j)$: samples from the dataset
- Becomes a "classical" ML compression problem.
- If too costly, can be approximated by Monte-Carlo.

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

LEFM Training Objective:

$$\mathcal{L}_{\text{LEFM}}(\theta) = \mathbb{E}_{\substack{x_0 \sim p_0 \\ x_1 \sim \hat{p}_{\text{data}} \\ t \sim U([0,1]) \\ b(2),\dots,b(M) \sim \hat{p}_{\text{data}}}} \left[\| u_\theta(x_t, t) - \hat{u}^\star_M(x_t, t) \|^2 \right]$$

with

$$x_t = (1-t)x_0 + t x_1, \quad b(1) := x_1,$$

$$\hat{u}^\star_M(x,t) = \sum_{j=1}^{M} \lambda(x,t) \frac{b(j) - x}{1-t}$$

$$\text{where } \lambda(x,t) = \text{softmax}\Big[-\frac{\|x - t\, b(l)\|^2}{2(1-t)^2} \Big]_{l=1,\dots,M}$$

- Almost deterministic target

On the Closed-Form of Flow Matching: Generalization Does Not Arise from Target Stochasticity

Quentin Bertrand, Anne Gagneux, Mathurin Massias, Rémi Emonet

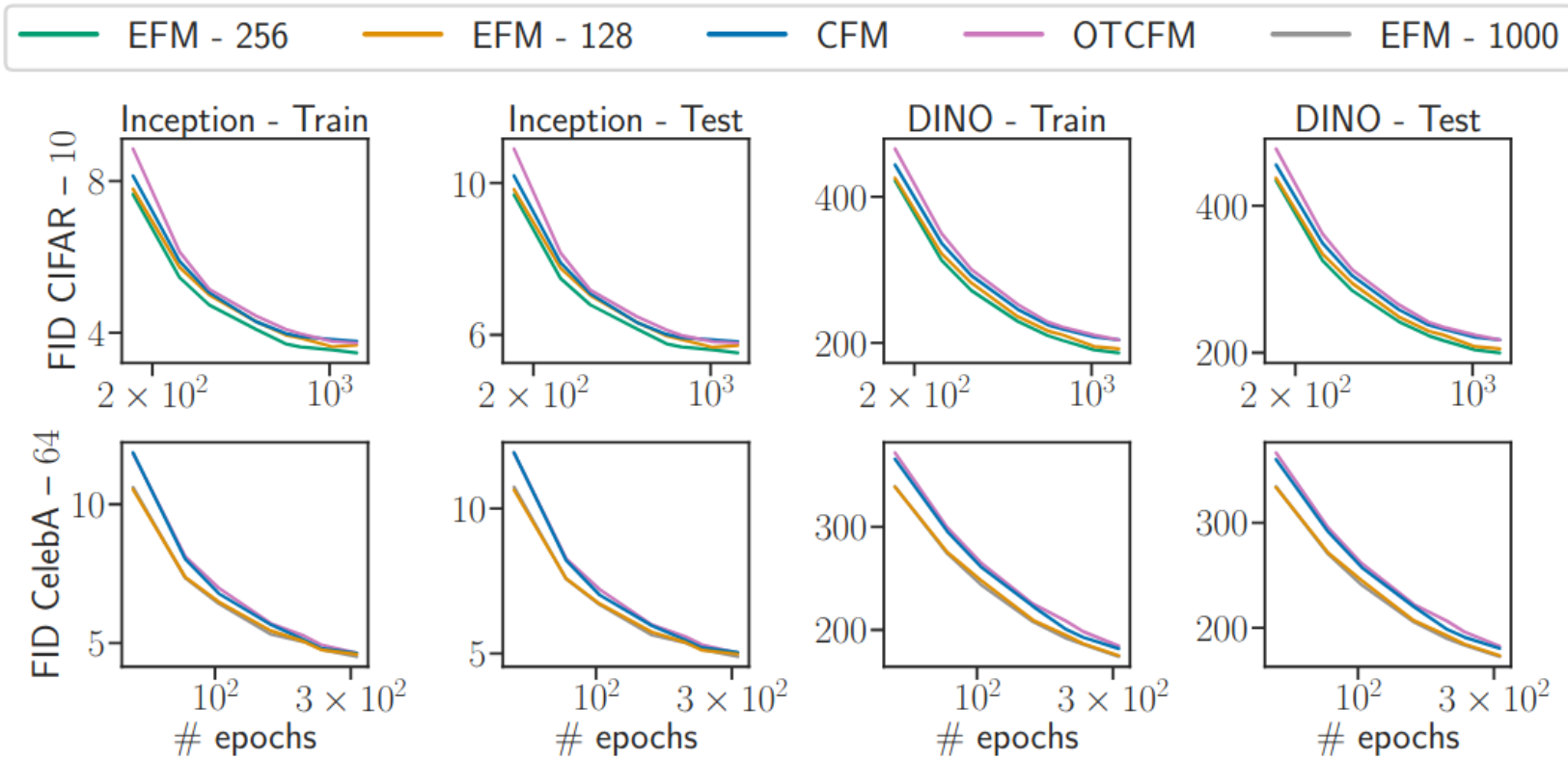


Figure 4: FID computed on the training set (50k) and the test set (10k) using multiple embeddings, Inception and DINOv2. Regressing against a more deterministic target (EFM - 128, 256, 1000) does not yield performance decreases. On the contrary, the more deterministic the target, the better the performance.

FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, Liang-Chieh Chen

Coarse-to-Fine Generation:

- Instead of generating the whole image at once, start with a very coarse version

FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, Liang-Chieh Chen

Coarse-to-Fine Generation:

- Instead of generating the whole image at once, start with a very coarse version
- Generate a higher-resolution image conditioned on the upsampled coarse version

FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching

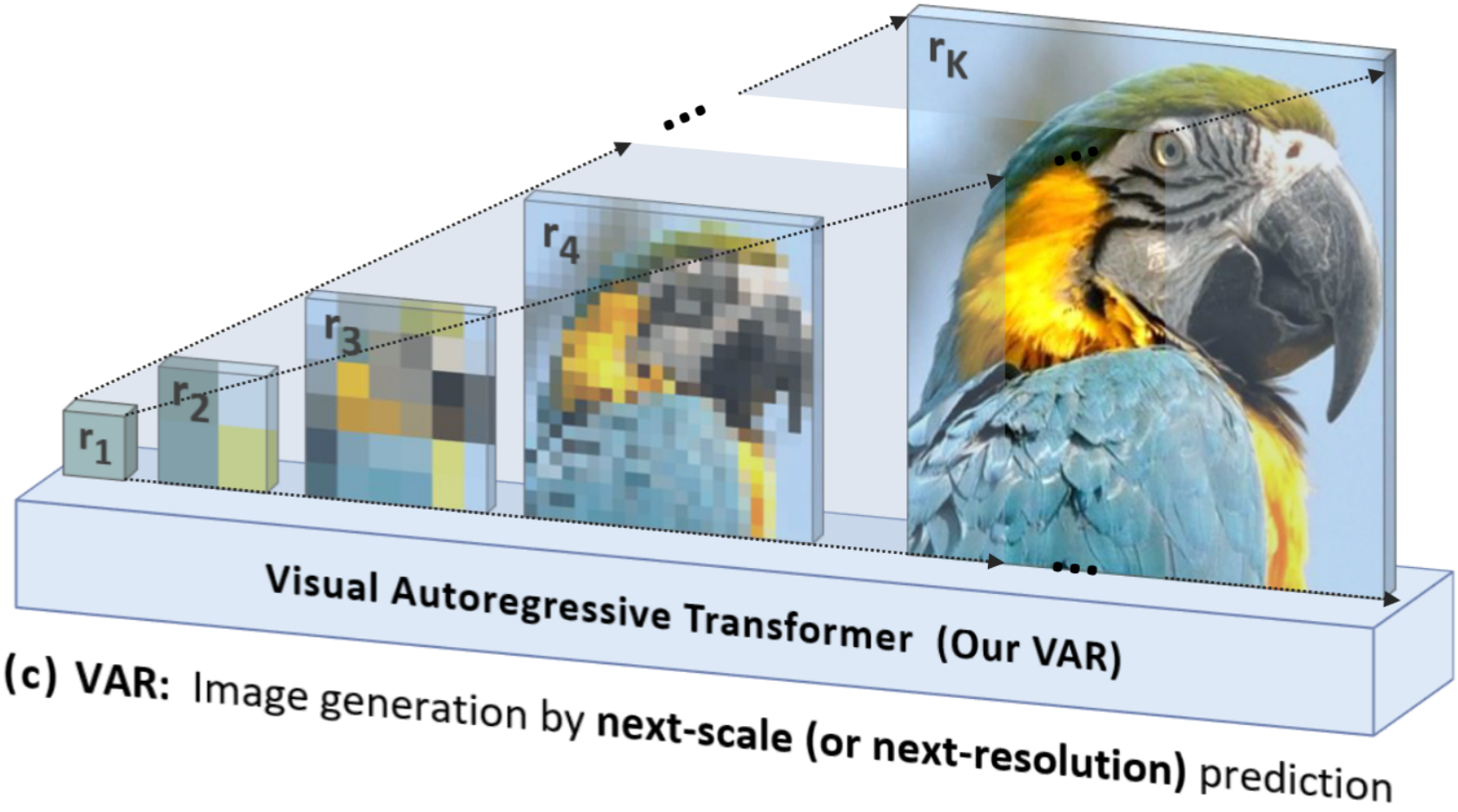
Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, Liang-Chieh Chen

Coarse-to-Fine Generation:

- Instead of generating the whole image at once, start with a very coarse version
- Generate a higher-resolution image conditioned on the upsampled coarse version
- Repeat the process to progressively refine the image

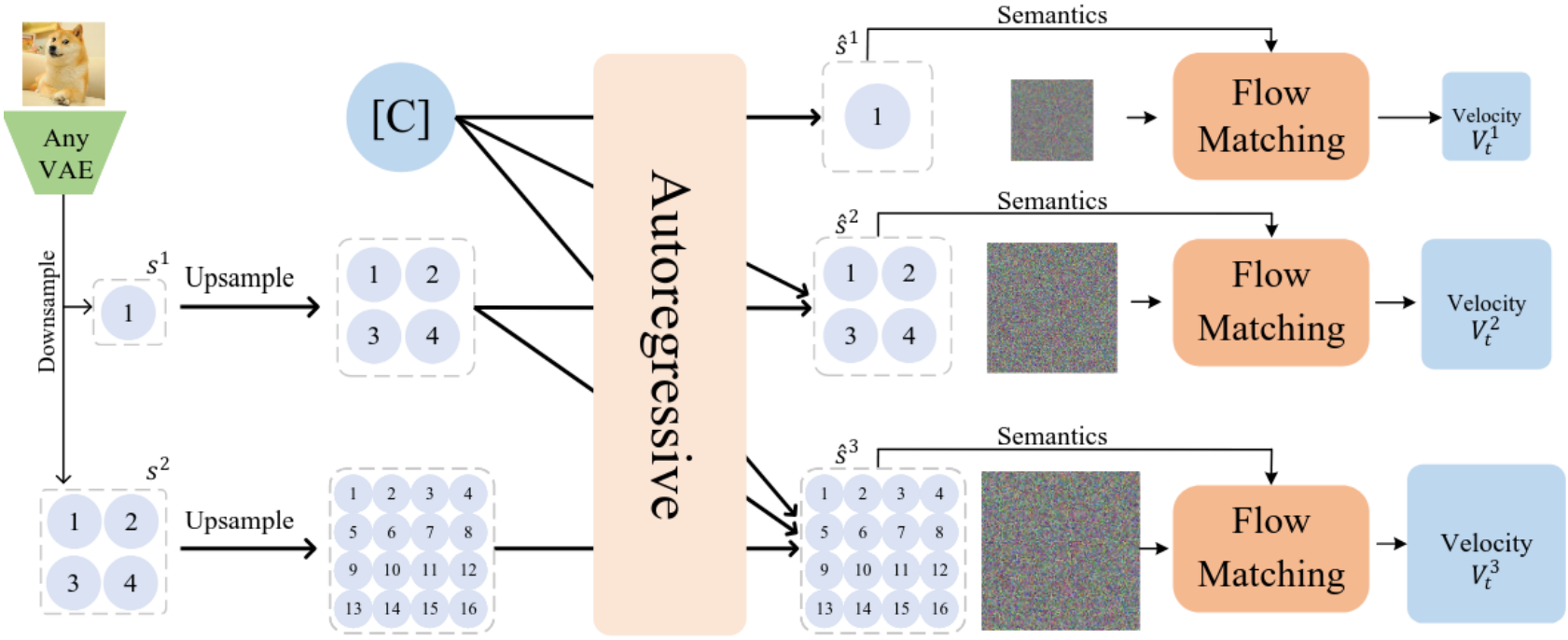
FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, Liang-Chieh Chen



FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, Liang-Chieh Chen



FlowAR: Scale-wise Autoregressive Image Generation Meets Flow Matching

Sucheng Ren, Qihang Yu, Ju He, Xiaohui Shen, Alan Yuille, Liang-Chieh Chen

method	type	params	FID ↓	IS ↑	Pre.↑	Rec.↑
BigGAN [6]	GAN	112M	6.95	224.5	0.89	0.38
GigaGAN [20]	GAN	569M	3.45	225.5	0.84	0.61
StyleGAN [40]	GAN	166M	2.30	265.1	0.78	0.53
ADM [12]	diffusion	554M	10.94	101.0	0.69	0.63
LDM [37]	diffusion	400M	3.60	247.7	-	-
U-ViT[5]	diffusion	287M	3.40	219.9	0.83	0.52
DiT [31]	diffusion	675M	2.27	278.2	0.83	0.57
SiT [2]	flow matching	675M	2.06	270.3	0.82	0.59
VQVAE-2 [36]	token-wise autoregressive	13.5B	31.11	45.0	0.36	0.57
VQGAN [14]	token-wise autoregressive	1.4B	15.78	74.3	-	-
RQTransformer [23]	token-wise autoregressive	3.8B	7.55	134.0	-	-
LlamaGen-B [42]	token-wise autoregressive	111M	5.46	193.6	0.83	0.45
ViT-VQGAN [51]	token-wise autoregressive	1.7B	4.17	175.1	-	-
LlamaGen-L [42]	token-wise autoregressive	343M	3.07	256.1	0.83	0.52
LlamaGen-XL [42]	token-wise autoregressive	775M	2.62	244.1	0.80	0.57
LlamaGen-3B [42]	token-wise autoregressive	3.1B	2.18	263.3	0.81	0.58
VAR-d12 [44]	scale-wise autoregressive	132M	5.81	201.3	0.81	0.45
FlowAR-S	scale-wise autoregressive	170M	3.61	234.1	0.83	0.50
VAR-d16 [44]	scale-wise autoregressive	310M	3.55	280.4	0.84	0.51
FlowAR-B	scale-wise autoregressive	300M	2.90	272.5	0.84	0.54
VAR-d20 [44]	scale-wise autoregressive	600M	2.95	302.6	0.83	0.56
FlowAR-L	scale-wise autoregressive	589M	1.90	281.4	0.83	0.57
VAR-d30 [44]	scale-wise autoregressive	2.0B	1.97	323.1	0.82	0.59
FlowAR-H	scale-wise autoregressive	1.9B	1.65	296.5	0.83	0.60

Conclusion

Thanks for your attention.