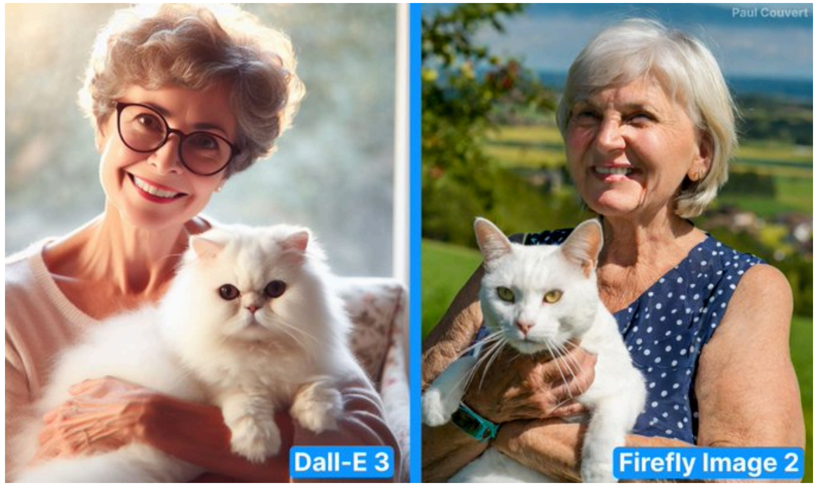
Generative Modeling By Rectified Flow
--Concepts and Frontier

刘星超

Contents

Background-AIGC



Images

Text-to-Video generation: "a horse galloping on a street"

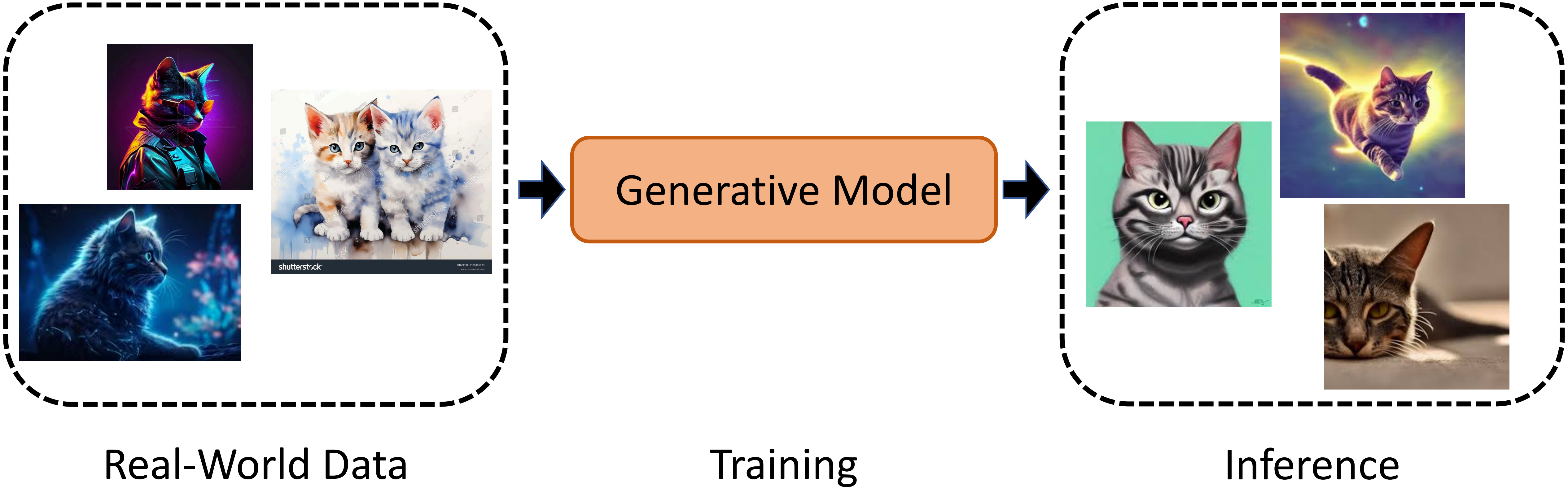
Text-to-Video generation: "a panda is playing guitar on times square"

Videos

Texts & Codes

Policies

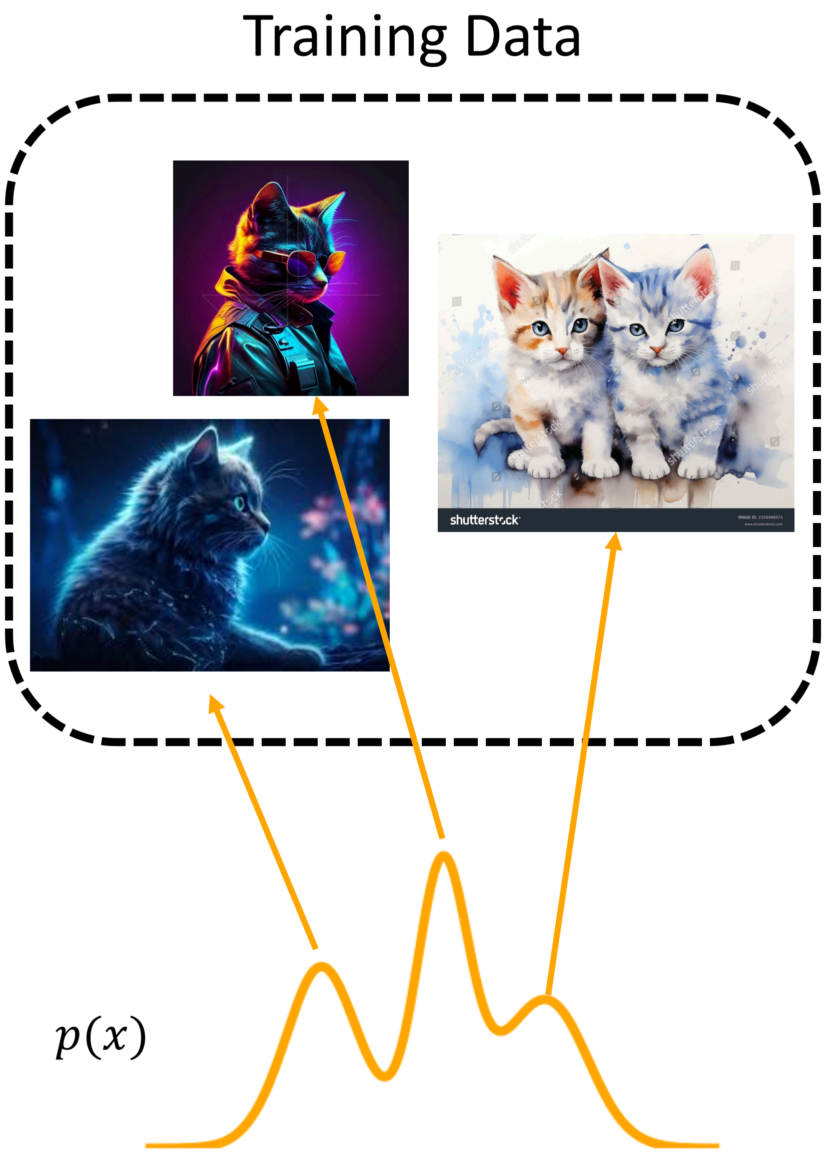
Pipeline of AIGC



Generative Models

Given: observed data points $\{x_i\}_{i=1}^n$

Unknown: the groundtruth data distribution $p(x)$



Generative Models

Given: observed data points $\{x_i\}_{i=1}^{n}$

Unknown: the groundtruth data distribution $p(x)$

Training: to learn a **model** to capture $p(x)$

Sampling: generate from the learned distribution

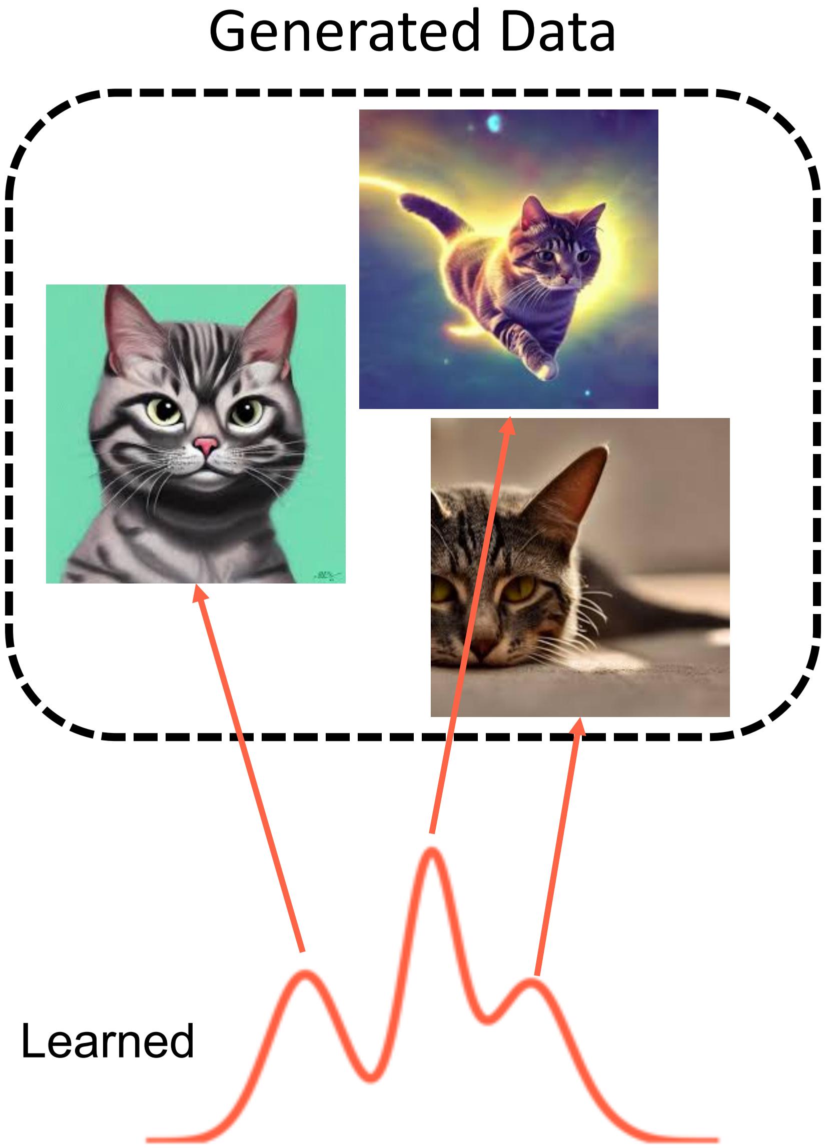
A Good Generative Model

=

Good for training

+

Good for sampling



Generative Models

Good old methods

Normalizing Flow

GAN

VAE

EBM

- Complicated math
- Mediocre quality
- GAN and VAE have fast sampling
- Training them is HARD

Generative Models

Good old methods

Normalizing Flow

GAN

VAE

EBM

- Complicated math
- Mediocre quality
- GAN and VAE have fast sampling
- Training them is HARD

Modern methods

Autoregressive

- Simple math
- Great quality
- Slow sampling
- Stable training

Diffusion

- Complicated math
- Great quality
- Slow sampling
- Stable training

Generative Models

Good old methods

Modern methods

Normalizing

VAE

Diffusion

- Co
- Me
- GAN and VAE have fast sampling
- Training them is HARD

- nplicated math
- at quality
- Slow sampling
- Stable training

- Slow sampling
- Stable training

Don’t we deserve something that is:

1. Simple math
2. Great quality
3. Fast sampling
4. Stable training?

Life Made Simple by Rectified Flow

- Simple math

  -- See right

- Great quality

  -- FLUX, Kling…

- Fast sampling

  -- One-step

- Stable training

DDPM

$$
\begin{aligned}
L_{\mathrm{VLB}} &= \mathbb{E}_{q(\mathbf{x}_{0:T})}\Big[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_\theta(\mathbf{x}_{0:T})}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{\prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_T)\prod_{t=1}^T p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=1}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \Big(\frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} \cdot \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}\Big) + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[-\log p_\theta(\mathbf{x}_T) + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} + \log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{q(\mathbf{x}_1|\mathbf{x}_0)} + \log \frac{q(\mathbf{x}_1|\mathbf{x}_0)}{p_\theta(\mathbf{x}_0|\mathbf{x}_1)}\Big] \\
&= \mathbb{E}_q\Big[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)\Big] \\
&= \mathbb{E}_q[\underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_T))}_{L_T} + \sum_{t=2}^T \underbrace{D_{\mathrm{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \underbrace{- \log p_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}]
\end{aligned}
$$

Rectified Flow

$$L = \int_0^1 \mathbb{E}_{\substack{X_0\sim\text{noise},\\ X_1\sim\text{data}}} [\| (X_1 - X_0) - v(X_t, t) \|^2]\, \mathrm{d}t,$$

$$\text{with } X_t = t\, X_1 + (1-t) X_0$$

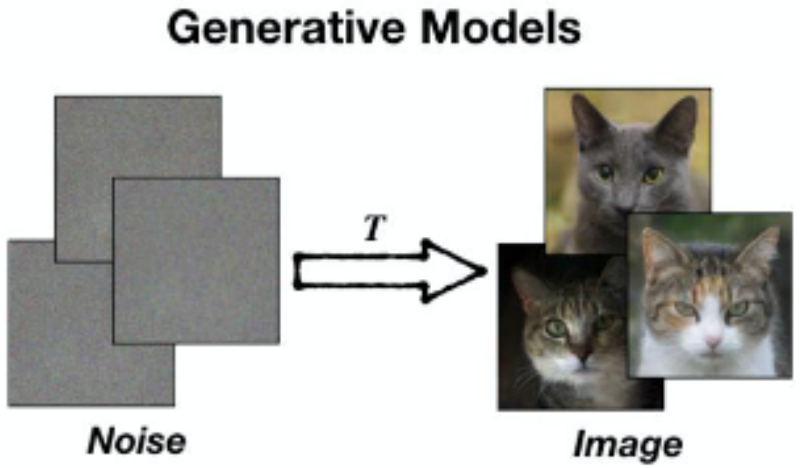
Rectified Flow – Problem of Interest

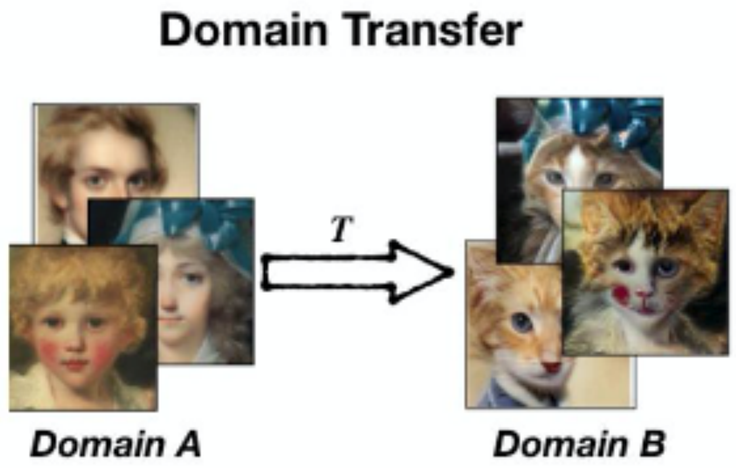
Given: observed data points from two distributions

$$\{x_i^0\}_{i=1}^n \sim \pi_0, \quad \{x_i^1\}_{i=1}^n \sim \pi_1$$

Goal: find a transport map T such that,

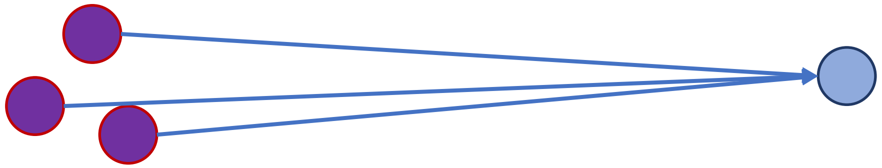
$$Z_1 \coloneqq T(Z_0) \sim \pi_1 \text{ when } Z_0 \sim \pi_0$$





Rectified Flow – How?

One-step generation is not hard…
When your target distribution is simple



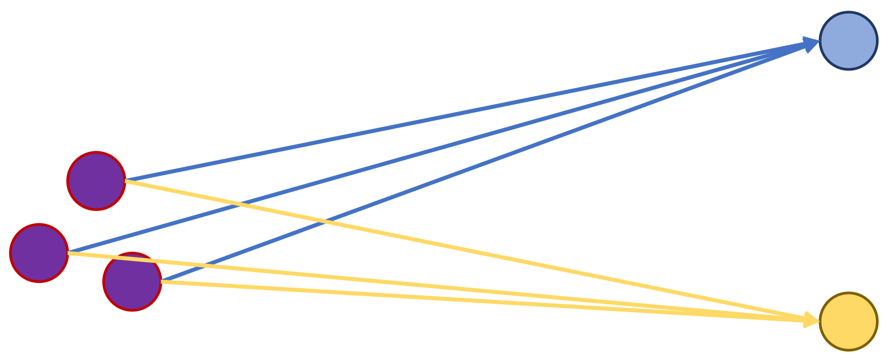
$\pi_0 \sim N(0, I)$

π_1: a single data point

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|T_\theta(X_0) - X_1\right|\right|^2$$

Rectified Flow – How?

It starts to get harder even for two points



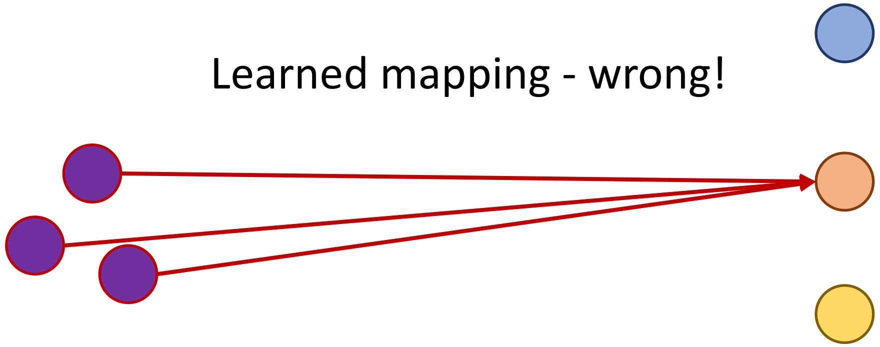
$\pi_0 \sim N(0, I)$

π_1: two data points

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|T_\theta(X_0) - X_1\right|\right|^2$$

Rectified Flow – How?

It starts to get harder even for two points



$\pi_0 \sim N(0, I)$

π_1: two data points

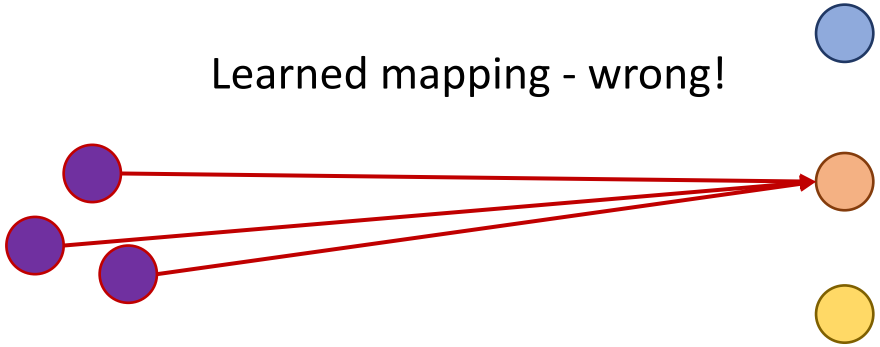
$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|T_\theta(X_0) - X_1\right|\right|^2$$

Rectified Flow – How?

It starts to get harder even for two points

Regression won't give good generative models

How to fix?



$\pi_0 \sim N(0, I)$

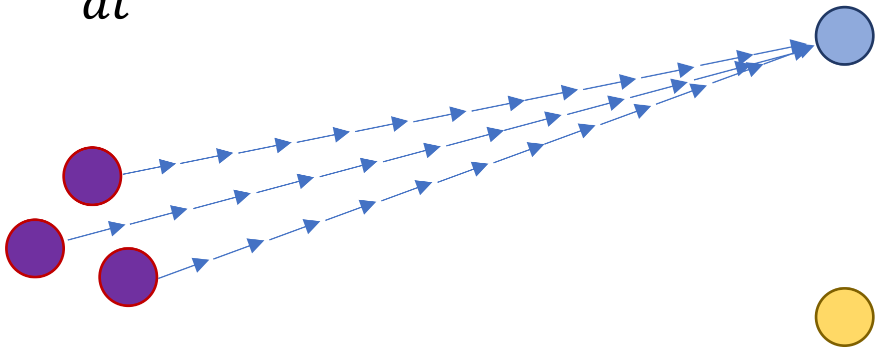
π_1: two data points

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|T_\theta(X_0) - X_1\right|\right|^2$$

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1 - t)X_0$



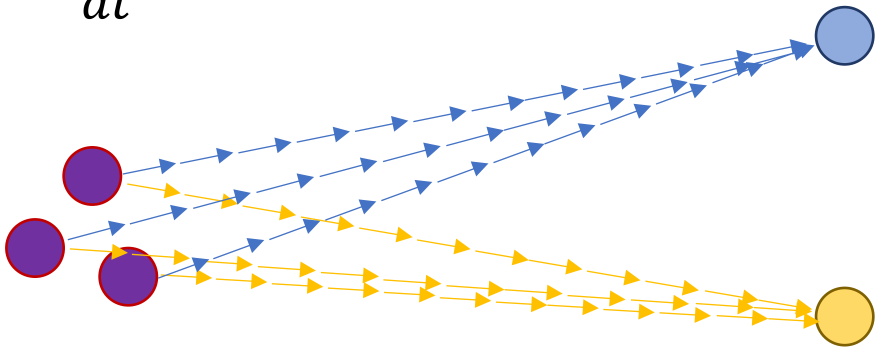
$\pi_0 \sim N(0, I)$

π_1: two data points

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



$\pi_0 \sim N(0, I)$

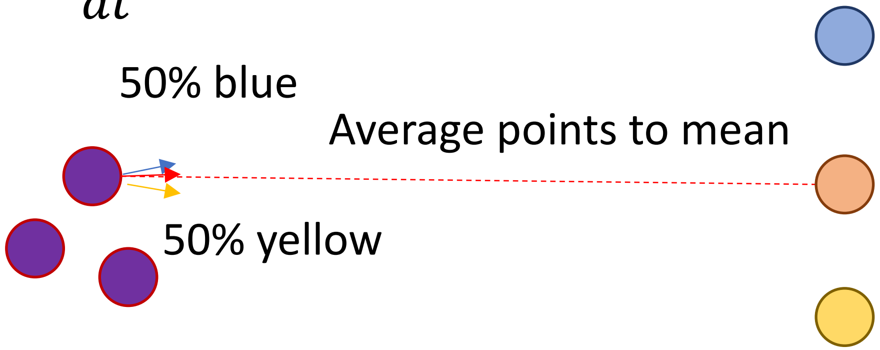
π_1: two data points

What if we average the velocity field instead?

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



$\pi_0 \sim N(0, I)$

π_1: two data points

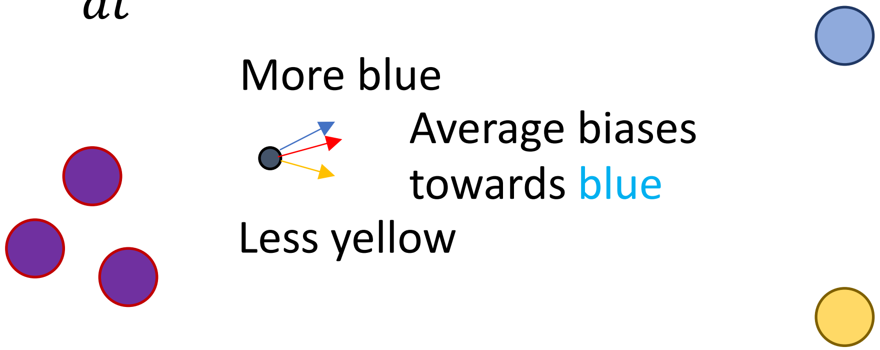
What if we average the velocity field instead?

t=0

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



$\pi_0 \sim N(0, I)$

π_1: two data points

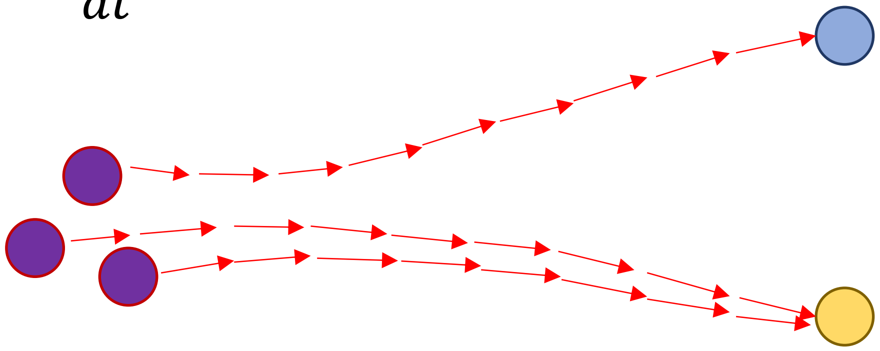
What if we average the velocity field instead?

t=0.1

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



$\pi_0 \sim N(0, I)$

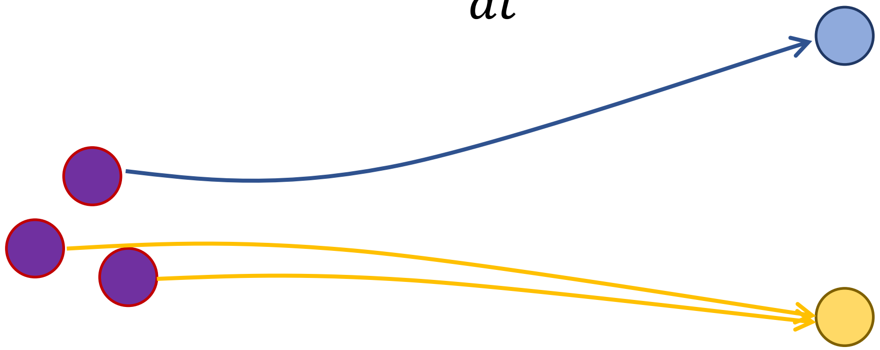
π_1: two data points

What if we average the velocity field instead?

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = v$



$\pi_0 \sim N(0, I)$ π_1: two data points

What if we average the velocity field instead?

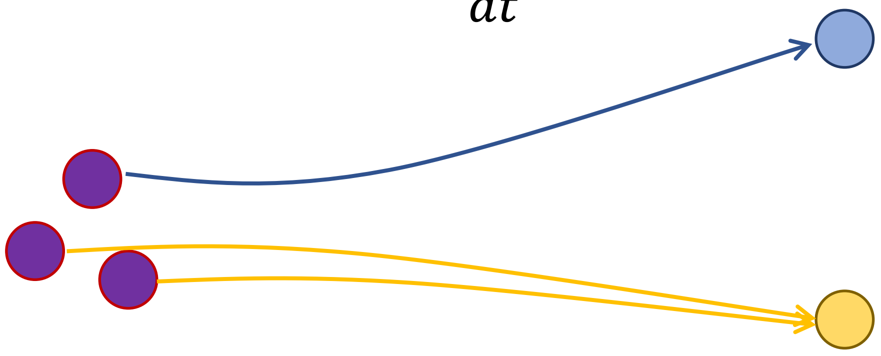
We get a vector field that maps to the correct distribution with flows

Please refer to paper for why it holds

Rectified Flow – How?

What if the mappings are now straight ODEs?

ODE: $\frac{dX}{dt} = v$



$\pi_0 \sim N(0, I)$

π_1: two data points

Let's learn a NN to copy this flow field!

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|v_\theta(X_t, t) - v(X_t, t)\right|\right|^2$$

Rectified Flow – How?

Let's learn a NN to copy this flow field!

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|v_\theta(X_t, t) - v(X_t, t)\right|\right|^2, where\ X_t = tX_1 + (1-t)X_0$$

$v(X_t, t)$ is the average of all the velocity vectors passing (X_t, t)

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left|v_\theta(X_t, t) - E_{X_0' \sim \pi_0, X_1' \sim \pi_1}[X_1' - X_0' | tX_1' + (1-t)X_0' = X_t]\right|\right|^2$$

Rectified Flow – How?

Let's learn a NN to copy this flow field!

$$\min_{\theta} E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - v(X_t,t)\right|\right|^2, where\ X_t = tX_1 + (1-t)X_0$$

$v(X_t,t)$ is the average of all the velocity vectors passing (X_t,t)

$$\min_{\theta} E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - E_{X_0'\sim\pi_0, X_1'\sim\pi_1}[X_1' - X_0' | tX_1' + (1-t)X_0' = X_t]\right|\right|^2$$

An equation that holds for L2

$$\min_{\theta} E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - (X_1 - X_0)\right|\right|^2$$

Rectified Flow – How?

Let's learn a NN to copy this flow field!

$$\min_{\theta} E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - v(X_t,t)\right|\right|^2, where\ X_t = tX_1 + (1-t)X_0$$

$v(X_t,t)$ is the average of all the velocity vectors passing (X_t,t)

$$\min_{\theta} E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - E_{X_0'\sim\pi_0, X_1'\sim\pi_1}[X_1' - X_0' | tX_1' + (1-t)X_0' = X_t]\right|\right|^2$$

An equation that holds for L2

$$\min_{\theta} E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - \boxed{(X_1 - X_0)}\right|\right|^2$$

Randomly select two points and compute the velocity

Rectified Flow – How?

Let's learn a NN to copy this flow field!

$$\min_\theta E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - v(X_t,t)\right|\right|^2, where\ X_t = tX_1 + (1-t)X_0$$

$\Downarrow$

$v(X_t, t)$ is the average of all the velocity vectors passing (X_t, t)

$$\min_\theta E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - E_{X_0'\sim\pi_0, X_1'\sim\pi_1}[X_1' - X_0' | tX_1' + (1-t)X_0' = X_t]\right|\right|^2$$

$\Downarrow$

An equation that holds for L2

$$\min_\theta E_{X_0\sim\pi_0, X_1\sim\pi_1} \left|\left|v_\theta(X_t,t) - \boxed{(X_1 - X_0)}\right|\right|^2$$

Randomly select two points and compute the velocity

Derivation is not related to the forms of π_0 and π_1
-Works for (under mild conditions) arbitrary π_0 and π_1!

Rectified Flow – Another View

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 \sim \pi_1} \left|\left| v_\theta(X_t, t) - \boxed{(X_1 - X_0)} \right|\right|^2$$

Randomly select two points and compute the velocity

$\{x_i^0\}_{i=1}^n \sim \pi_0$

2-GMM

$\{x_i^1\}_{i=1}^n \sim \pi_1$

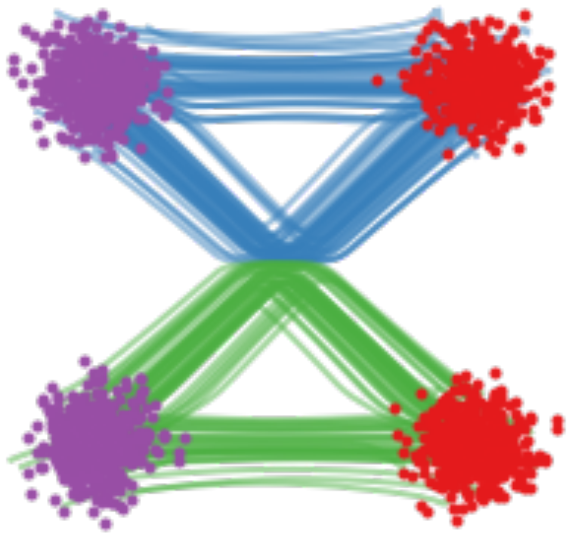
2-GMM

Training ODEs

Linear Interpolation: $X_t = tX_1 + (1-t)X_0$

ODE: $\frac{dX}{dt} = X_1 - X_0$

Learned Flow



Simulate with ODE solver, e.g., Euler

ODE: $\frac{dX}{dt} = v_\theta(X, t)$

Probability Flow Ordinary Differential Equation

Rectified Flow – Algorithm

- **Given:** $\{x_i^0\}_{i=1}^n \sim \pi_0$, $\{x_i^1\}_{i=1}^n \sim \pi_1$
- **Training Iteration (Batch size = 1)**:
  - Step 1: Randomly sample $X_0 \in \{x_i^0\}_{i=1}^n$ and $X_1 \in \{x_i^1\}_{i=1}^n$
  - Step 2: Randomly sample $t \in [0,1]$
  - Step 3: Compute gradient with loss

$$L(\theta) \coloneqq \left|\left|X_1 - X_0 - v_\theta(X_t, t)\right|\right|^2,$$

where $\quad X_t = tX_1 + (1-t)X_0$

Rectified Flow – Implementation

Algorithm 2 Train(Data)

```
# Input: Data={x0, x1}
# Output: Model v(x,t) for the rectified flow
initialize Model
for x0, x1 in Data:  # x0, x1: samples from π0, π1
    Optimizer.zero_grad()
    t = torch.rand(batchsize)  # Randomly sample t ∈ [0,1]
    Loss = ( Model(t*x1+(1-t)*x0, t) - (x1-x0) ).pow(2).mean()
    Loss.backward()
    Optimizer.step()
return Model
```

Algorithm 3 Sample(Model, Data)

```
# Input: Model v(x,t) of the rectified flow
# Output: draws of the rectified coupling (Z0, Z1)
coupling = []
for x0, _ in Data:  # x0: samples from π0 (batchsize×dim)
    x1 = model.ODE_solver(x0)
    coupling.append((x0, x1))
return coupling
```

CIFAR10

Method	NFE (↓)	IS (↑)	FID (↓)
VP SDE	2000	9.58	2.55
subVP SDE	2000	9.56	2.61
VP ODE	140	9.37	3.93
subVP ODE	146	9.46	3.16
Rectified Flow	**127**	**9.60**	**2.58**

Fast sampling + high-quality

(A) LSUN Church (B) CelebA HQ (C) LSUN Bedroom (D) AFHQ Cat

256 Resolution

Rectified Flow – Common Misunderstandings

1. Straight Line

2. Why does it avoid crossing?

Rectified Flow

Don't we deserve something that is:

1. Simple math
2. Great quality
3. Fast sampling 🤔
4. Stable training?

Rectified Flow - Sampling

- In computer, we solve ODEs by Euler discretization

$$X_{t+\epsilon} = X_t + \epsilon\, v(X_t, t)$$

ϵ: step size

Large ϵ: Fast, inaccurate ; Small ϵ: Accurate, slow

Rectified Flow - Sampling

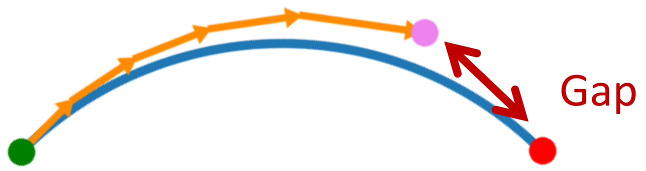
- In computer, we solve ODEs by Euler discretization

$$X_{t+\epsilon} = X_t + \epsilon\, v(X_t, t)$$

ϵ: step size

Large ϵ: Fast, inaccurate ; Small ϵ: Accurate, slow

Curved Trajectory



$N = 5, \epsilon = 1/N$

Straight Trajectory

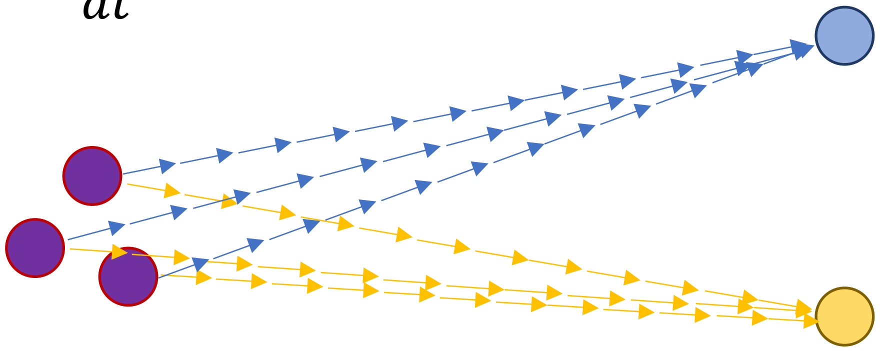
$N = 1, \epsilon = 1/N$

$$\mathrm{d}X = v(X, t)\mathrm{d}t$$

Rectified Flow - Reflow

Why not straight?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



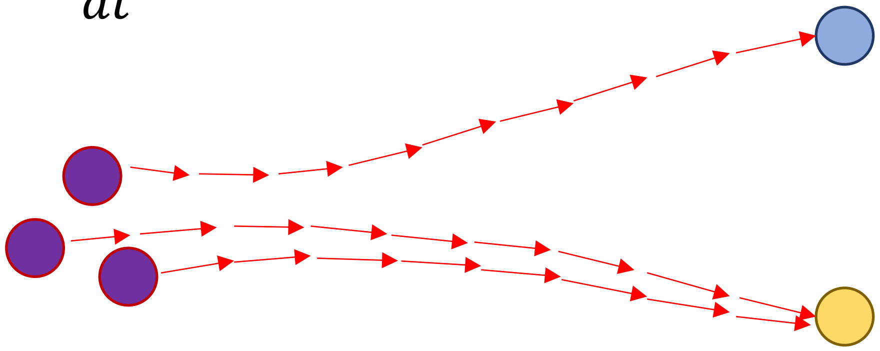
$\pi_0 \sim N(0, I)$

π_1: two data points

Rectified Flow – Reflow

What if we change the teacher?

ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



$\pi_0 \sim N(0, I)$

π_1: two data points

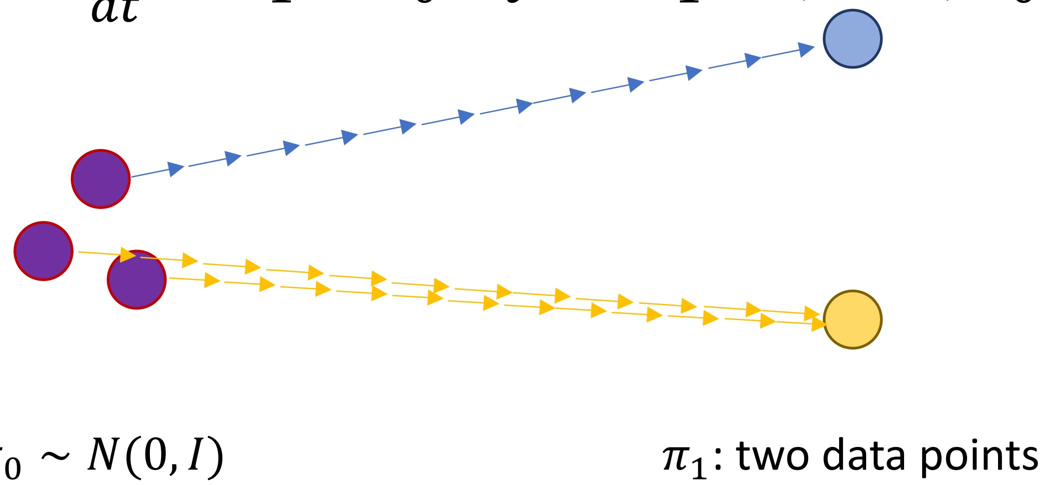
There is no more crossing in the learned flow!
Good coupling only!
Keeps the correct distribution π_1!

And thus keeps the math correct

Rectified Flow - Reflow

Construct new straight-line teachers

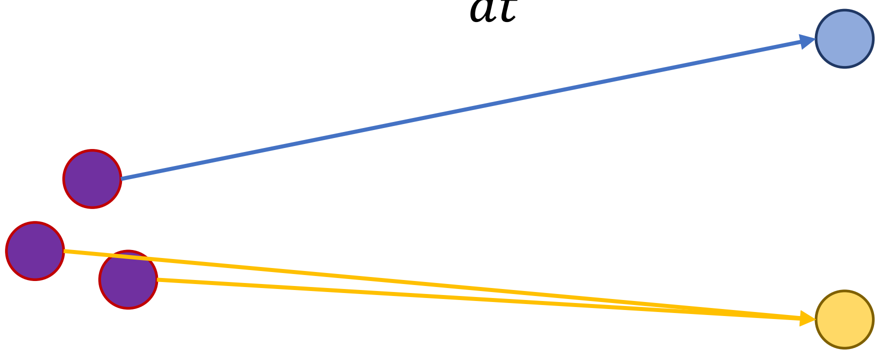
ODE: $\frac{dX}{dt} = X_1 - X_0, X_t = tX_1 + (1-t)X_0$



Rectified Flow - Reflow

New flow – straighter!

ODE: $\frac{dX}{dt} = v$



$\pi_0 \sim N(0, I)$

π_1: two data points

Learning NN by copying this new field gives faster model

$$\min_{\theta} E_{X_0 \sim \pi_0, X_1 = ODE_{v_{old}}(X_0)} \left|\left|v_\theta(X_t, t) - v(X_t, t)\right|\right|^2$$

Rectified Flow – Implementation of Reflow

Algorithm 4 Reflow(Data)

```
# Input:  Data={x0, x1}
# Output:  draws of the K-th rectified coupling
Coupling = Data
for k = 1,...,K:
    Model = Train(Coupling)
    Coupling = sample(Model, Data)
return Coupling
```

Algorithm 2 Train(Data)

```
# Input:  Data={x0, x1}
# Output:  Model v(x,t) for the rectified flow
initialize Model
for x0, x1 in Data:  # x0, x1: samples from π0, π1
    Optimizer.zero_grad()
    t = torch.rand(batchsize)  # Randomly sample t ∈ [0,1]
    Loss = ( Model(t*x1+(1-t)*x0, t) - (x1-x0) ).pow(2).mean()
    Loss.backward()
    Optimizer.step()
return Model
```

Rectified Flow – Difference Between Distillation & Reflow

Distillation

$$\min_{\phi} \mathrm{E}_{X_0 \sim \pi_0, X_1 = ODE_{v_{old}}(X_0)} \left\| f_\phi(X_0) - X_1 \right\|^2$$

- Output: One-step generation
- Not invertible

Reflow

$$\min_{\theta} \int_0^1 \mathrm{E}_{X_0 \sim \pi_0, X_1 = ODE_{v_{old}}(X_0)} \left[\left| \left| (X_1 - X_0) - v_\theta(X_t, t) \right| \right|^2 \right] \mathrm{d}t$$

- Output: Straighter Flow, Any-step generation
- Invertible
- The resulting flow is more friendly for distillation

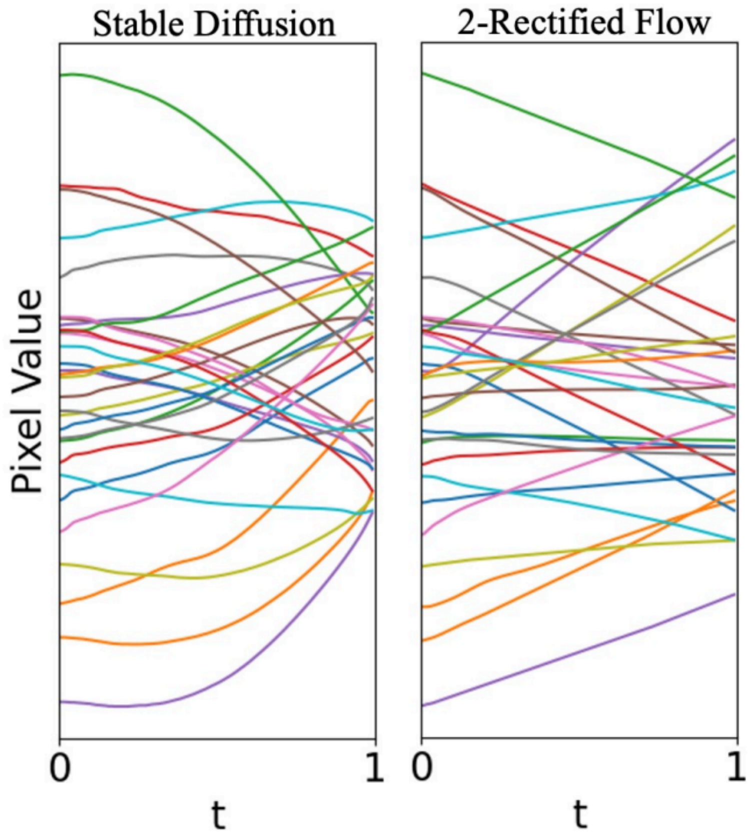
CIFAR10

Method	NFE (↓)	IS (↑)	FID (↓)
1-Rectified Flow	127	9.60	2.58
2-Rectified Flow	110	9.24	3.36
3-Rectified Flow	104	9.01	3.96

Method	NFE (↓)	IS (↑)	FID (↓)
1-Rectified Flow	1	1.13	378
2-Rectified Flow	1	8.08	12.21
3-Rectified Flow	1	8.47	8.15

Method	NFE (↓)	IS (↑)	FID (↓)
1-Rectified Flow+Distill	1	9.08	6.18
2-Rectified Flow+Distill	1	9.01	$\boxed{4.85}$
3-Rectified Flow+Distill	1	8.79	5.21

SOTA
(when arXiv)

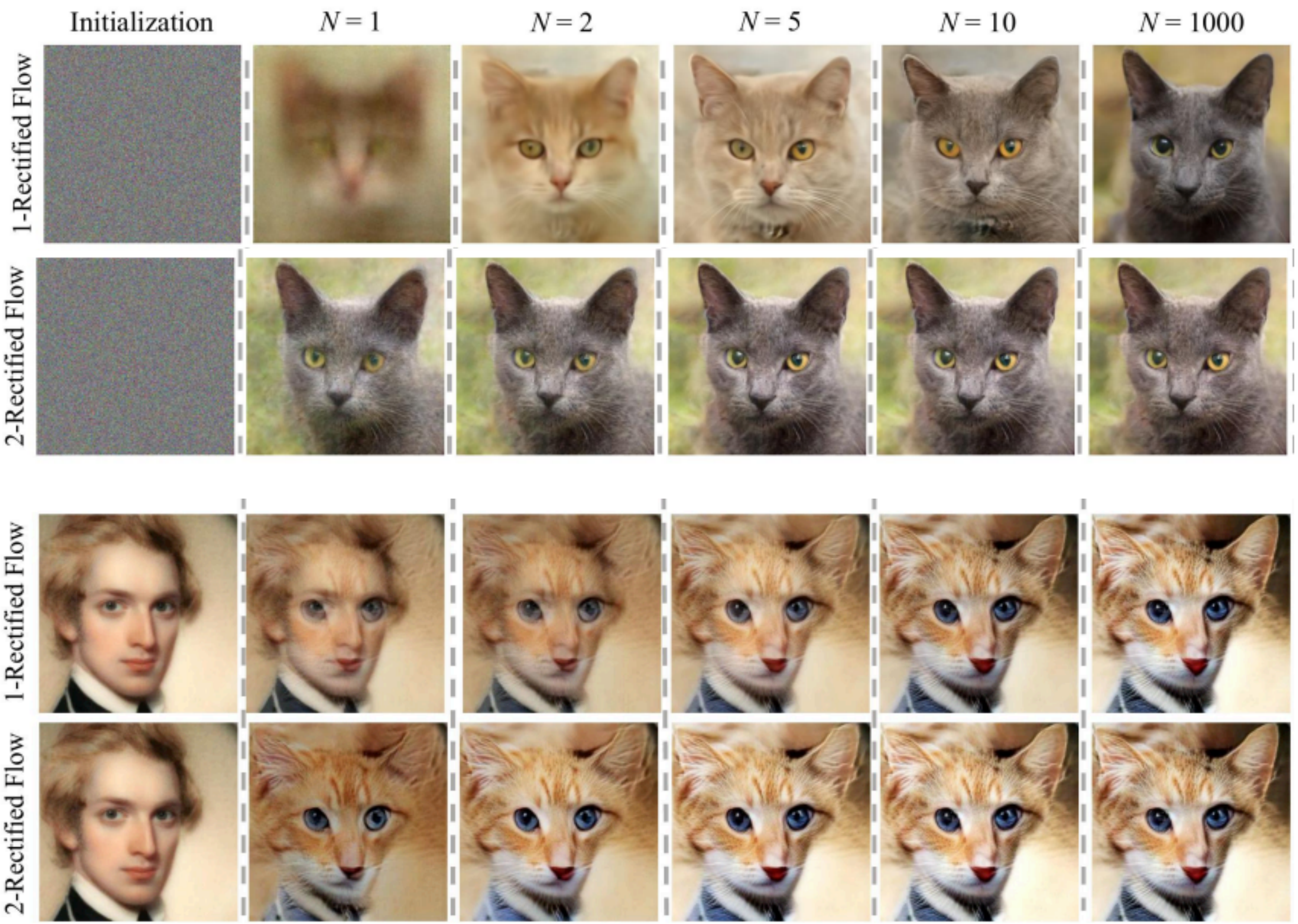

Stable Diffusion
2-Rectified Flow
Pixel Value
0
1
t
0
1
t

Rectified Flow

Don't we deserve something that is:

1. Simple math ✅
2. Great quality ✅
3. Fast sampling ✅
4. Stable training? ✅

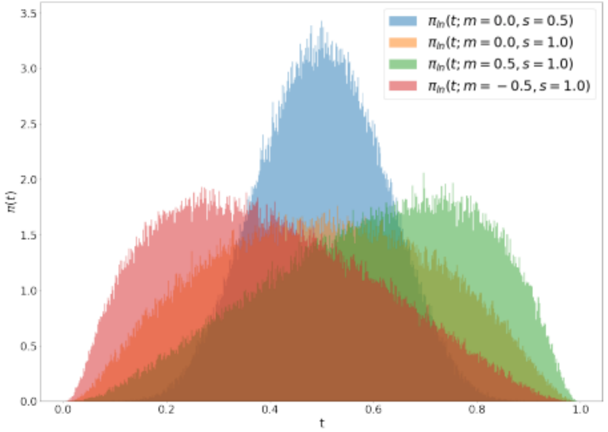
And so is the correctness of RF!



Initialization
$N = 1$
$N = 2$
$N = 5$
$N = 10$
$N = 1000$
1-Rectified Flow
2-Rectified Flow
1-Rectified Flow
2-Rectified Flow

Questions?

Applications – Stable Diffusion 3(, Kling…)



	rank averaged over		
variant	all	5 steps	50 steps
rf/lognorm(0.00, 1.00)	1.54	1.25	1.50
rf/lognorm(1.00, 0.60)	2.08	3.50	2.00
rf/lognorm(0.50, 0.60)	2.71	8.50	1.00
rf/mode(1.29)	2.75	3.25	3.00
rf/lognorm(0.50, 1.00)	2.83	1.50	2.50
eps/linear	2.88	4.25	2.75
rf/mode(1.75)	3.33	2.75	2.75
rf/cosmap	4.13	3.75	4.00
edm(0.00, 0.60)	5.63	13.25	3.25
rf	5.67	6.50	5.75
v/linear	6.83	5.75	7.75
edm(0.60, 1.20)	9.00	13.00	9.00
v/cos	9.17	12.25	8.75
edm/cos	11.04	14.25	11.25
edm/rf	13.04	15.25	13.25
edm(-1.20, 1.20)	15.58	20.25	15.00

Table 1. **Global ranking of variants.** For this ranking, we apply non-dominated sorting averaged over EMA and non-EMA weights, two datasets and different sampling settings.

Prove the effectiveness of RF through EXTENSIVE experiments

	ImageNet		CC12M	
variant	CLIP	FID	CLIP	FID
rf	0.247	49.70	0.217	94.90
edm(-1.20, 1.20)	0.236	63.12	0.200	116.60
eps/linear	0.245	48.42	0.222	*90.34*
v/cos	0.244	50.74	0.209	97.87
v/linear	0.246	51.68	0.217	100.76
rf/lognorm(0.50, 0.60)	**0.256**	80.41	<u>0.233</u>	120.84
rf/mode(1.75)	*0.253*	**44.39**	0.218	94.06
rf/lognorm(1.00, 0.60)	<u>0.254</u>	114.26	**0.234**	147.69
rf/lognorm(-0.50, 1.00)	0.248	<u>45.64</u>	0.219	**89.70**
rf/lognorm(0.00, 1.00)	0.250	*45.78*	*0.224*	<u>89.91</u>

Table 2. **Metrics for different variants.** FID and CLIP scores of different variants with 25 sampling steps. We highlight the **best**, <u>second best</u>, and *third best* entries.

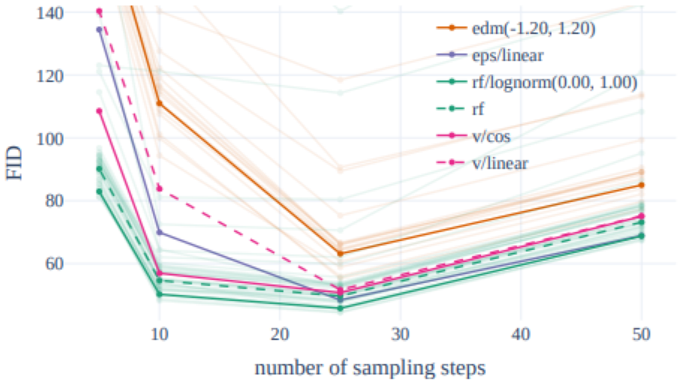
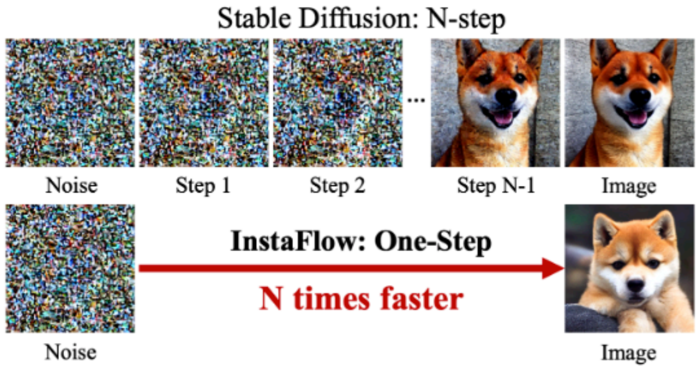
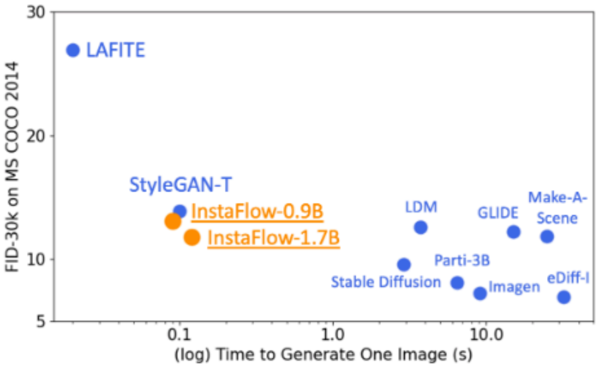


Figure 3. **Rectified flows are sample efficient.** Rectified Flows perform better then other formulations when sampling fewer steps. For 25 and more steps, only `rf/lognorm(0.00, 1.00)` remains competitive to `eps/linear`.

Applications – InstaFlow

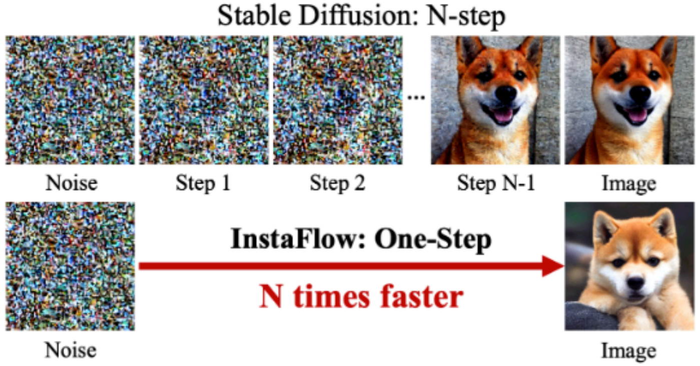
Will the rectified flow pipeline still work in Stable Diffusion level?

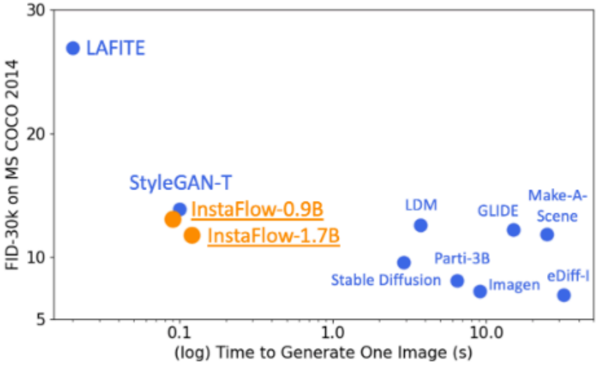




Applications – InstaFlow

Will the rectified flow pipeline still work in Stable Diffusion level?





- **Text-Conditioned Reflow**:

Random text from text dataset

Text-conditioned model

$$v_{k+1} = \arg\min_{v} \mathbb{E}_{X_0\sim\pi_0, \boxed{\mathcal{T}\sim D_{\mathcal{T}}}} \left[\int_0^1 || (X_1 - X_0) - \boxed{v(X_t, t \mid \mathcal{T})} ||^2 \,\mathrm{d}t\right],$$

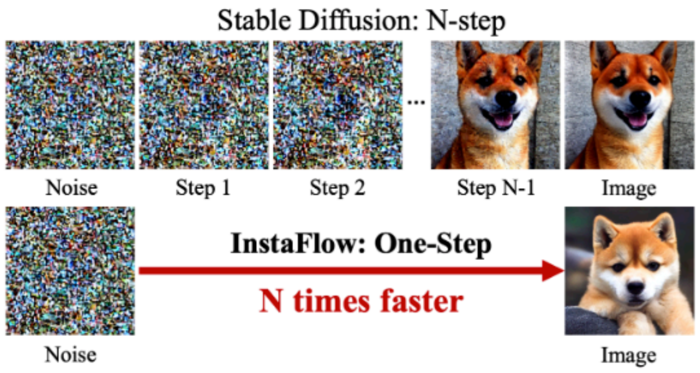
$$\text{with } \boxed{X_1 = \texttt{ODE}[v_k](X_0 \mid \mathcal{T})} \quad \text{and} \quad X_t = tX_1 + (1-t)X_0,$$

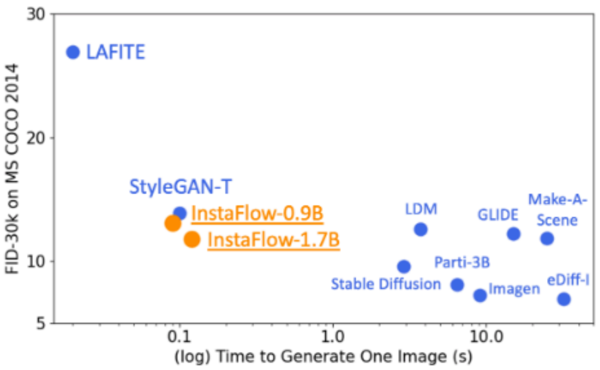
Text-conditioned generation

- **Text Dataset**: 1.6M data points from LAION-2B (aesthetics score 6.0+)
- **Model**: Stable Diffusion (as 1-Rectified Flow)
- **Training cost**: 199 A100 GPU days (InstaFlow 0.9B)

Applications – InstaFlow

Will the rectified flow pipeline still work in Stable Diffusion level?





- **Text-Conditioned Reflow**:

Random text from text dataset

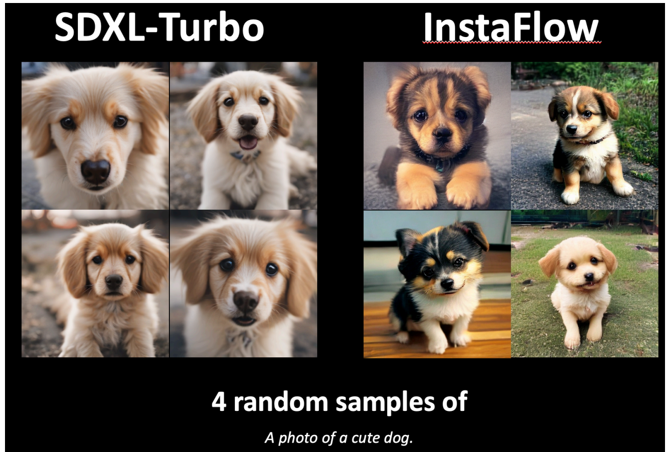
Text-conditioned model

$$v_{k+1} = \arg\min_{v} \mathbb{E}_{X_0\sim\pi_0,\boxed{\mathcal{T}\sim D_{\mathcal{T}}}}\left[\int_0^1 \|\,(X_1 - X_0) - \boxed{v(X_t, t \mid \mathcal{T})}\,\|^2\,\mathrm{d}t\right],$$

$$\text{with } \boxed{X_1 = \mathtt{ODE}[v_k](X_0 \mid \mathcal{T})} \text{ and } X_t = tX_1 + (1-t)X_0,$$

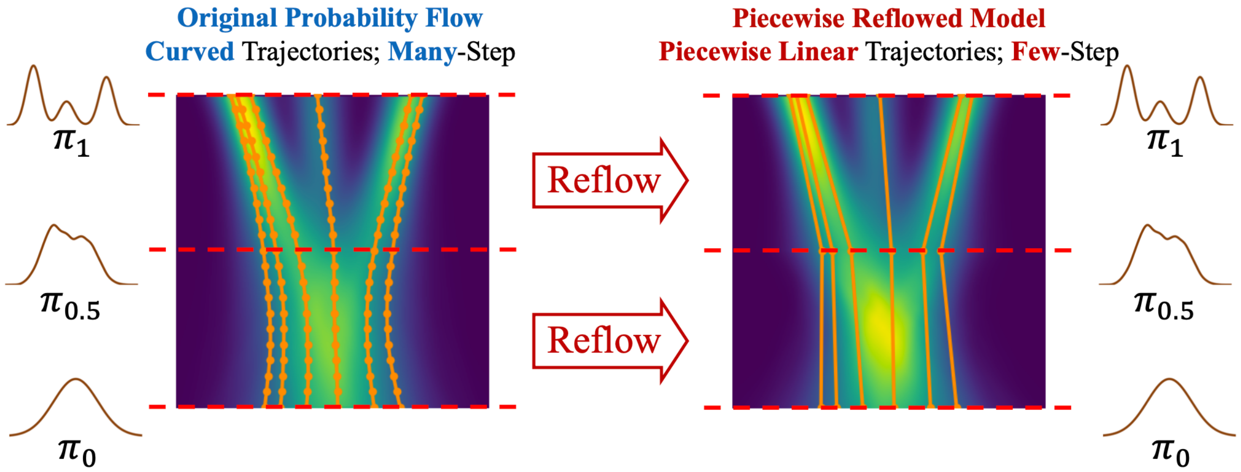
Text-conditioned generation

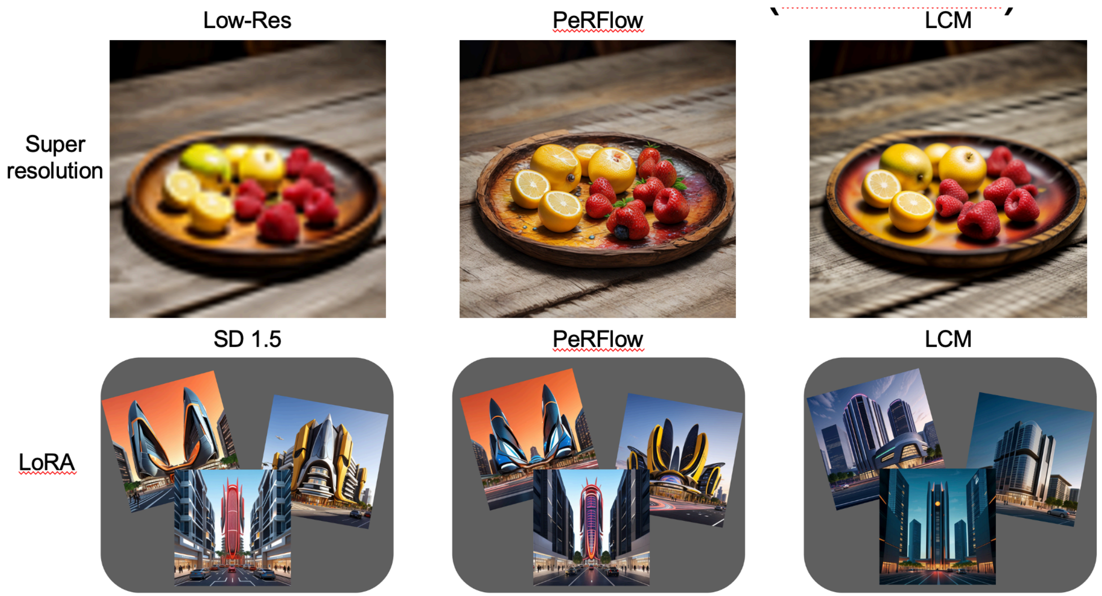
- **Text Dataset**: 1.6M data points from LAION-2B (aesthetics score 6.0+)
- **Model**: Stable Diffusion (as 1-Rectified Flow)
- **Training cost**: 199 A100 GPU days (InstaFlow 0.9B)



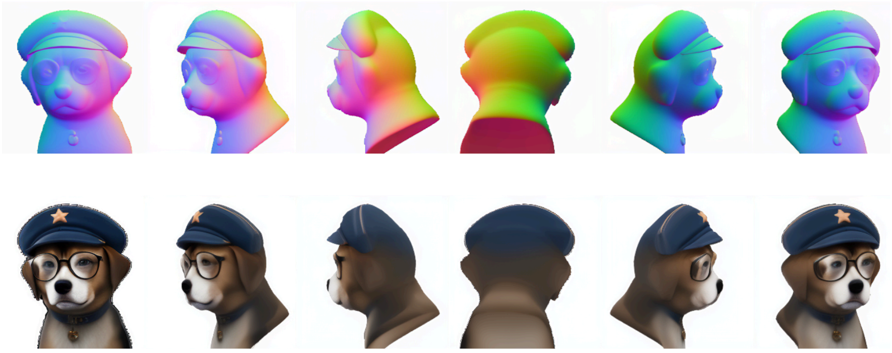
Applications – PeRFlow

Generating 1.6M data triplets is expensive. How to avoid that?





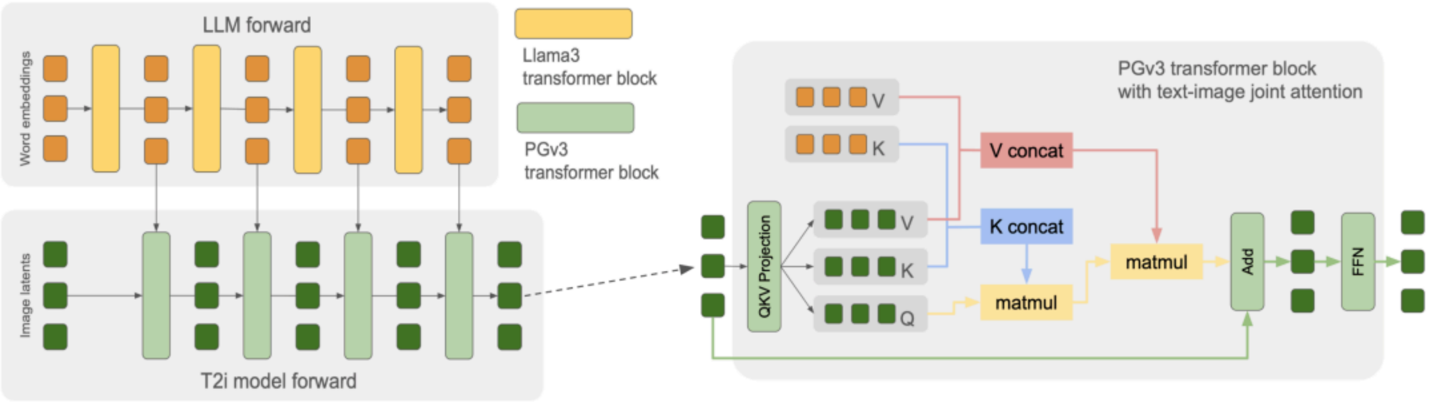
4-step



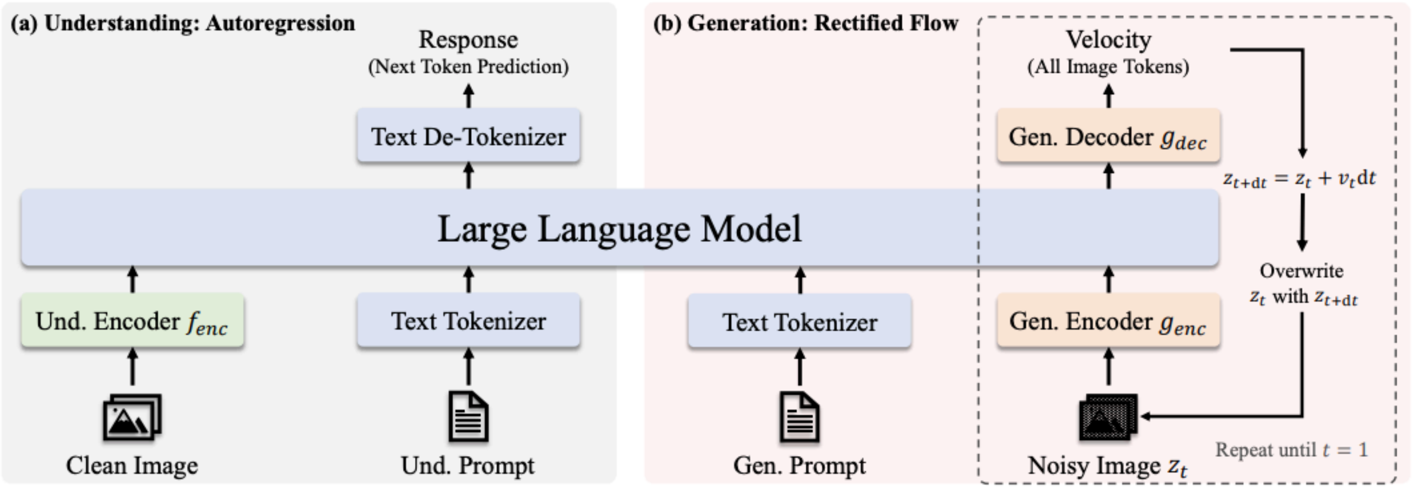
One-Step Wonder3D

Applications – JanusFlow

Can we unify text-to-image model?



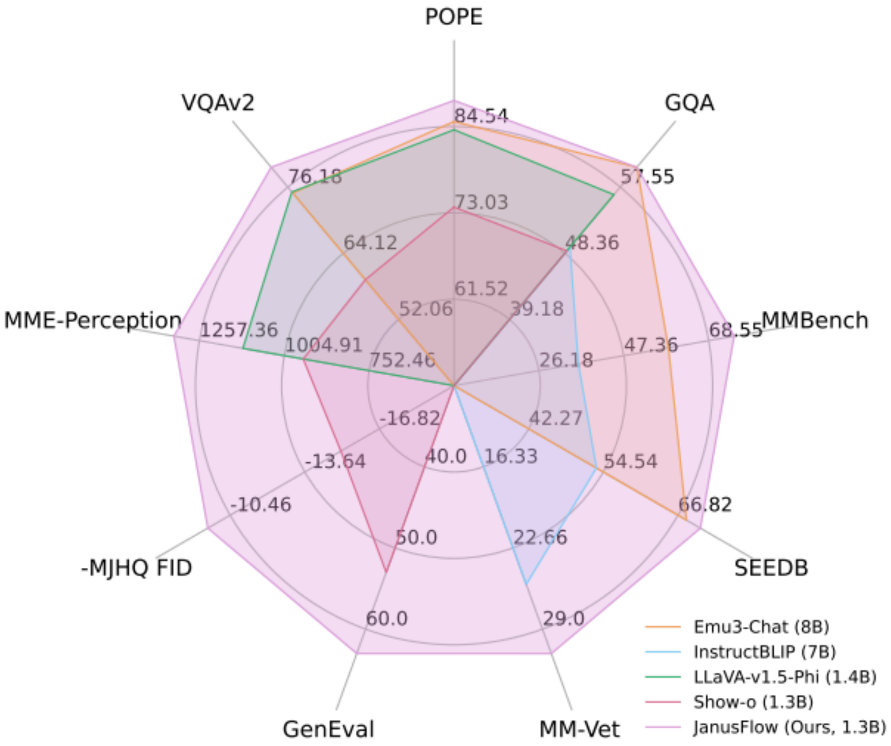
Playground v3 and all the previous works



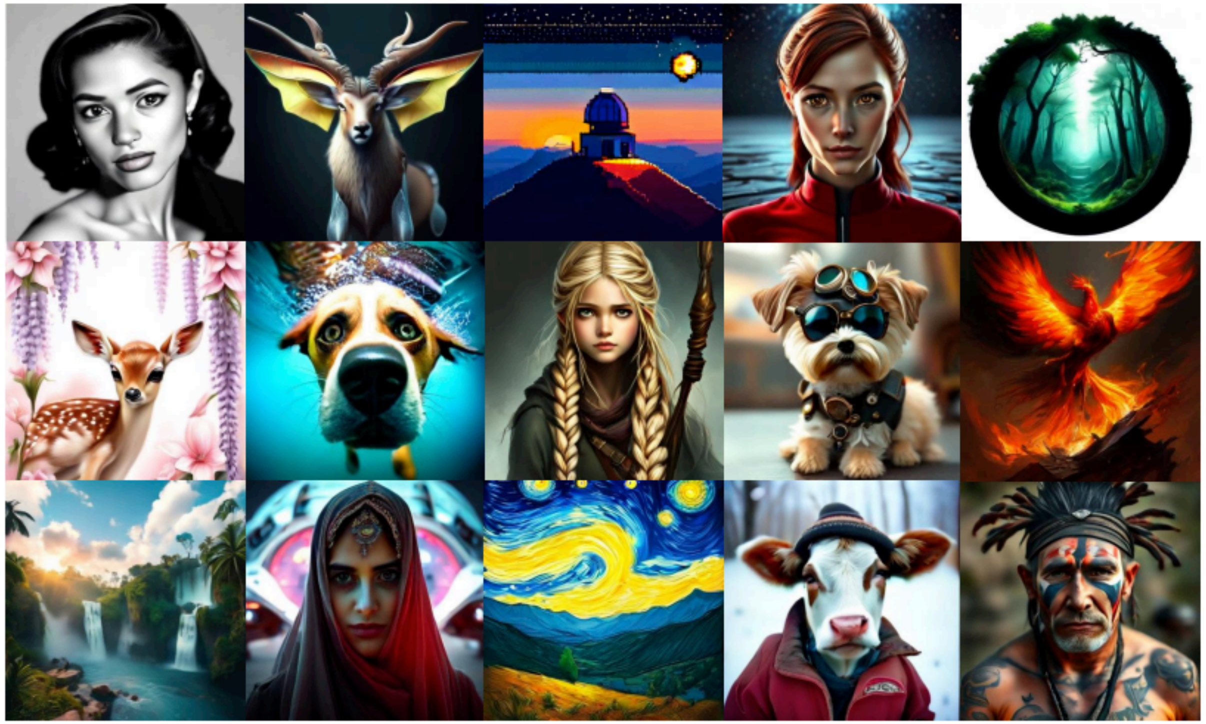
JanusFlow

LLMs are so magical

Applications – JanusFlow



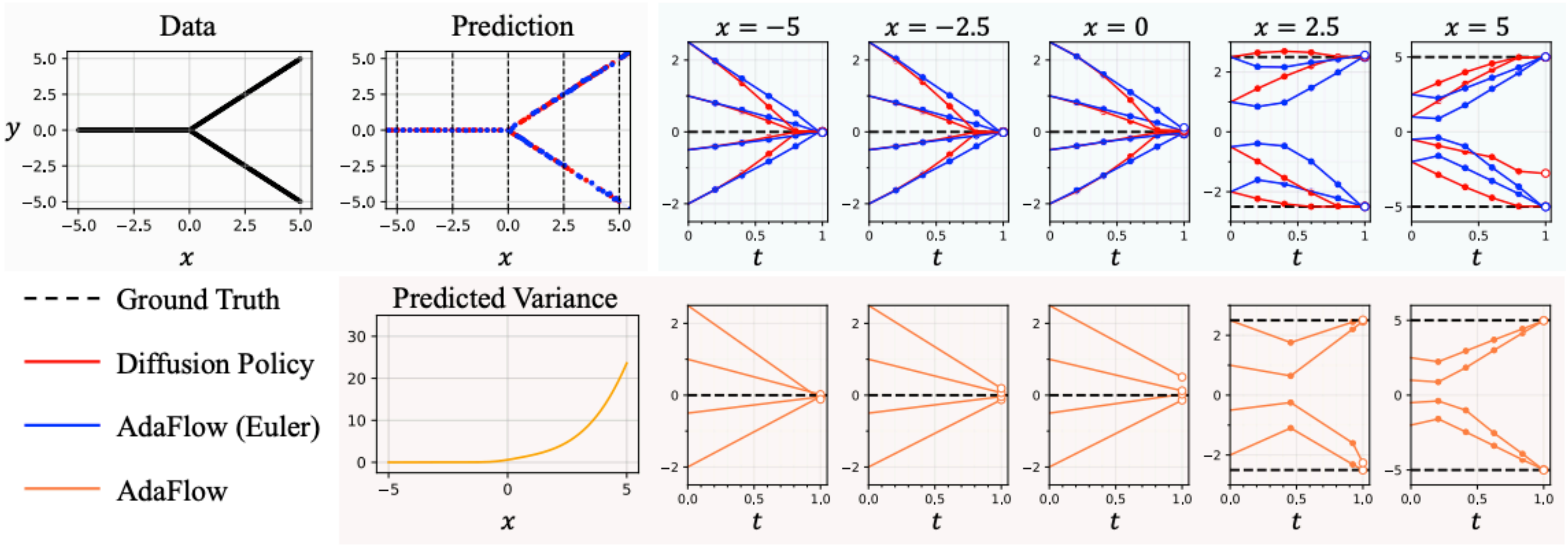
(a) Benchmark Performances.



(b) Visual Generation Results.

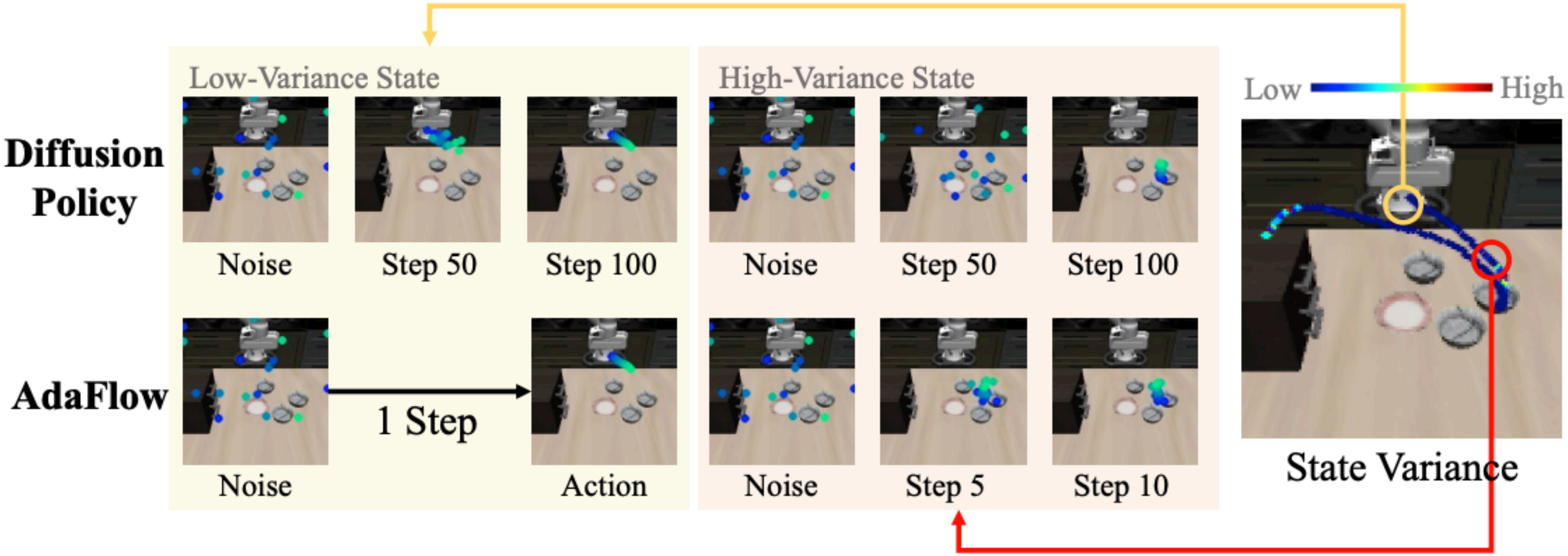
Applications – AdaFlow

We can also extend the RF to imitation learning to control robots
An adaptive policy for faster inference – Real-time is important!

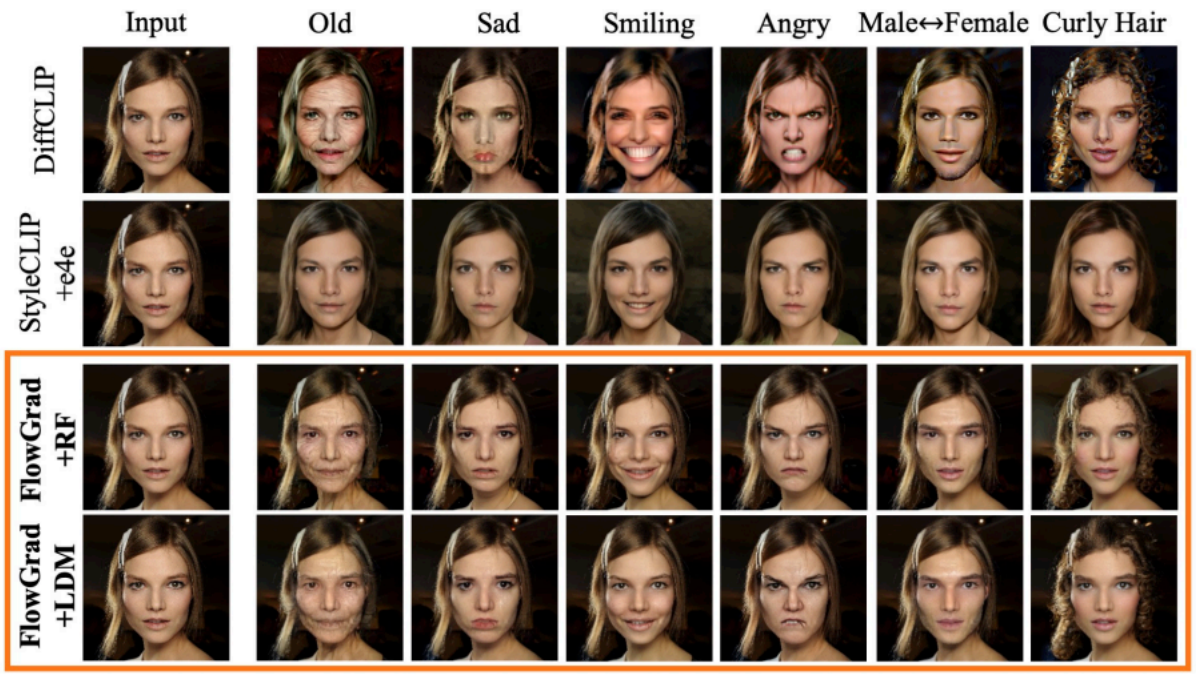


Applications – AdaFlow

We can also extend the RF to imitation learning to control robots
An adaptive policy for faster inference – Real-time is important!



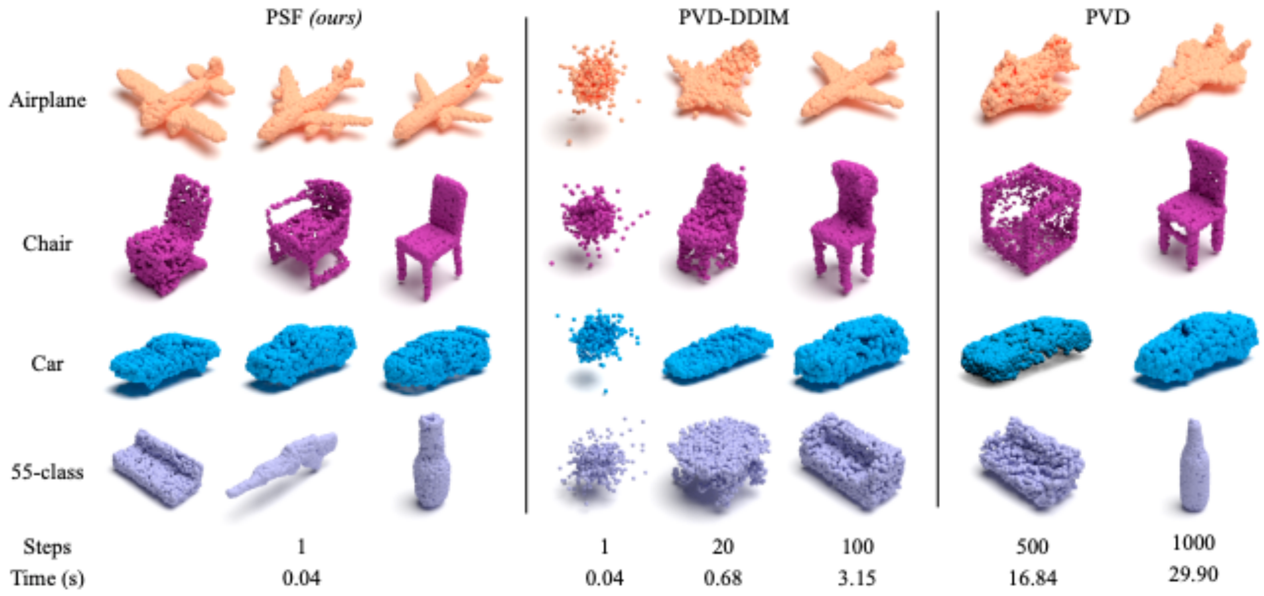
Applications – More



FlowGrad

Fast gradient-based editing with probability flows

[**Liu** et al., CVPR 2023]



Point Straight Flow

One-step point cloud generation (**100**× faster)

[Wu et al., CVPR 2023]

RF <-> GANs

Similarity	Difference
Goal: Matching noises with data	Explicit vs Implicit
Sampling: Fast	Any-step generation vs One-step
	Training: Supervised vs Adversarial
	Likelihood vs Non-Likelihood

RF <-> VAEs

Similarity	Difference
Encoder-Decoder	Continuous vs One-step
Gaussian matching	Explicit vs Implicit
	Training: Supervised vs Highly noisy
	High-quality vs Over-smoothed

RF <-> Normalizing Flows

Similarity

RFs are continuous normalizing flows

Not MLE, RF is a new paradigm for training CNFs

Difference

Eliminate the unscalable designs in training CNFs with MLE

RF <-> Diffusion Models

Similarity

Diffusion models can be transformed to RFs

Mathematically, DMs are equivalent to Gaussian RFs

Likelihood models

Difference

RFs are not restricted to Gaussian π_0

Reflow

RF way is much easier to understand & practical advantages!

RF <-> Autoregressive Models

Similarity

RF sampling is actually autoregressive

$$X_{t+\epsilon} = X_t + \epsilon\, v(X_t, t)$$

Even Markovian…

Difference

Implementation

RF <-> Consistency Models

Similarity

Sampling: Fast

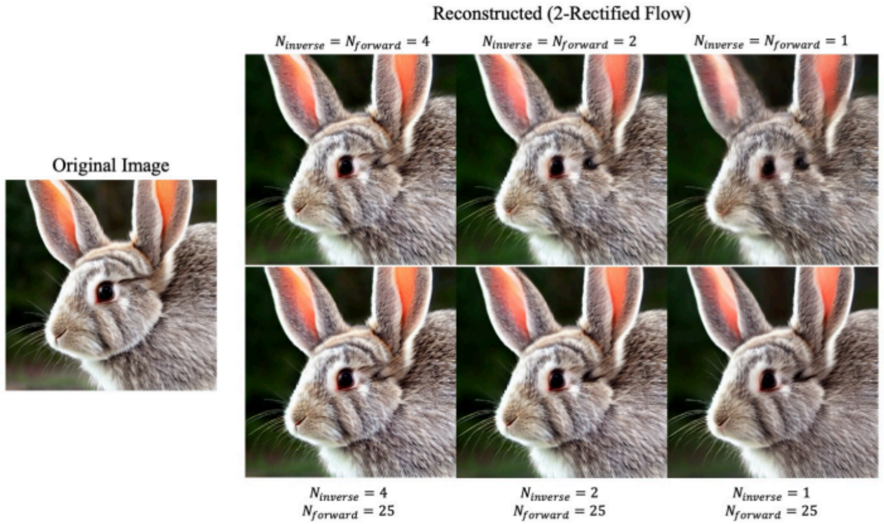
Relationship with DMs

Orthogonal Methods! You can train consistency models from RF

Difference

Sampling Convergence

Invertible vs Non-invertible (Likelihood vs Non-likelihood)



References

1. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow, ICLR 2023
   Xingchao Liu, Chengyue Gong, Qiang Liu
2. InstaFlow: One Step is Enough for High-Quality Diffusion-Based Text-to-Image Generation, ICLR 2024
   Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, Qiang Liu
3. PeRFlow: Piecewise Rectified Flow as Universal Plug-and-Play Accelerator, NeurIPS 2024
   Hanshu Yan, Xingchao Liu, Jiachun Pan, Jun Hao Liew, Qiang Liu, Jiashi Feng
4. AdaFlow: Imitation Learning with Variance-Adaptive Flow-Based Policies, NeurIPS 2024
   Xixi Hu, Bo Liu, Xingchao Liu, Qiang Liu
5. JanusFlow: Harmonizing Autoregression and Rectified Flow for Unified Multimodal Understanding and Generation
   Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai yu, Liang Zhao, Yisong Wang, Jiaying Liu, Chong Ruan
6. FlowGrad: Controlling the Output of Generative ODEs with Gradients, CVPR 2023
   Xingchao Liu, Lemeng Wu, Shujian Zhang, Chengyue Gong, Wei Ping, Qiang Liu
7. Fast Point Cloud Generation with Straight Flows, CVPR 2023
   Lemeng Wu, Dilin Wang, Chengyue Gong, Xingchao Liu, Yunyang Xiong, Rakesh Ranjan, Raghuraman Krishnamoorthi, Vikas Chandra, Qiang Liu

Many thanks to my collaborators

GitHub

Rectified Flow: https://github.com/gnobitab/RectifiedFlow

InstaFlow: https://github.com/gnobitab/InstaFlow

PeRFlow: https://github.com/magic-research/piecewise-rectified-flow

AdaFlow: https://github.com/hxixixh/adaflow

FlowGrad: https://github.com/gnobitab/FlowGrad

Point Straight Flow: https://github.com/klightz/PSF

JanusFlow: https://github.com/deepseek-ai/Janus

Thank you!

Questions?