Diff-Instruct: A Universal Approach for Transferring Knowledge From Pre-trained Diffusion Models

Weijian Luo[1], Tianyang Hu[2], Shifeng Zhang[2], Jiacheng Sun[2], Zhenguo Li[2], Zhihua Zhang[1]

Peking University[1], Huawei Noah's Ark Lab[2]

TL; DR: Unified Framework for Understanding Knowledge Transferring of Diffusion Models.

Background

Diffusion Models DM learns score functions by minimizing:

$$\mathcal{J}(\theta) = \int_0^T \mathbb{E}_{p_0(x_0)p_{0t}(x_t'|x_0)}[\lambda(t)||\nabla_{x_t'} \log p_{0t}(x_t'|x_0) - s_\theta(x_t', t)||_2^2]\mathrm{d}t$$

Knowledge Transferring of DM

- After training, a DM learns rich knowledge about data distributions, making them valuable assets.
- Moreover, in many applications, high-quality data (such as 3D datasets) is expensive (or even impossible) to obtain;
- we are motivated to study transferring knowledge of pre-trained diffusion models to other generative models, such as implicit generative models or generators.
- Diff-Instruct is tailored for such a knowledge transfer.

Our Work: The Diff-Instruct

Brief Summary of Diff-Instruct

- Set up a Math foundation of knowledge transferring of DMs;
- SOTA single-step diffusion distillation model;
- Consistently improving pre-trained GAN generator models;

Mathematical Setup

- We have a pre-trained teacher diffusion model $s_{p^{(t)}}(x_t)$ and a student implicit generative model g_θ, which can generate data samples efficiently through $x_0 = g_\theta(z), z \sim p_z$. We assume x_0 is differentiable to parameter θ.
- Let $p^{(t)}$ denote the underlying distribution of teacher DM, and $q^{(t)}$ the distribution of diffused student distribution.
- The goal is to minimize some divergence $\mathcal{D}(q^{(0)}, p^{(0)})$.

The Integral Kullback-Leibler divergence.

Definition. Given a forward diffusion process and a proper weighting function $w(t) > 0$, $t \in [0, T]$, the IKL divergence between two distributions p, q is defined as

$$\mathcal{D}_{IKL}^{[0,T]}(q, p) := \int_{t=0}^T w(t)\mathcal{D}_{KL}(q^{(t)}, p^{(t)})\mathrm{d}t := \int_{t=0}^T w(t)\mathbb{E}_{x_t\sim q^{(t)}}\big[\log \frac{q^{(t)}(x_t)}{p^{(t)}(x_t)}\big]\mathrm{d}t, \quad (1)$$

where $q^{(t)}$ and $p^{(t)}$ denote the marginal densities of the forward diffusion process at time t initialized with $q^{(0)} = q$ and $p^{(0)} = p$ respectively.

Diff-Instruct:

- The goal is to minimize the IKL divergence between $q^{(0)}$ and $p^{(0)}$
- The θ gradient of such an objective has a formula

$$\mathrm{Grad}(\theta) = \int_{t=0}^T w(t)\mathbb{E}_{\substack{z\sim p_Z, x_0=g_\theta(z),\\ x_t|x_0\sim p_t(x_t|x_0)}}\big[s_{q^{(t)}}(x_t, t) - s_{p^{(t)}}(x_t)\big]\frac{\partial x_t}{\partial \theta}\mathrm{d}t. \quad (2)$$

- But $s_{q^{(t)}}(x_t, t)$ is unknown, so we fine-tune an auxiliary diffusion model $s_\phi(x_t, t)$ with data consistently sampled from $q^{(0)}$ to approximate $s_{q^{(t)}}(x_t, t)$.

Diff-Instruct Algorithm:

Input: pre-trained DM $s_{q^{(t)}}$, student g_θ, prior distribution p_z, DM s_ϕ;

Repeat:

1. update ϕ using SGD with gradient:
   $\mathrm{Grad}(\phi) = \frac{\partial}{\partial \phi}\int_{t=0}^T w(t)\mathbb{E}_{\substack{x_0=g(\theta),\\ x_t|x_0\sim p_t(x_t|x_0)}}\|s_\phi(x_t, t) - \nabla_{x_t}\log p_t(x_t|x_0)\|_2^2\mathrm{d}t.$
2. update θ using SGD with the gradient:
   $\mathrm{Grad}(\theta) = \int_{t=0}^T w(t)\mathbb{E}_{\substack{x_0=g(\theta),\\ x_t|x_0\sim p_t(x_t|x_0)}}\big[s_\phi(x_t, t) - s_{p^{(t)}}(x_t)\big]\frac{\partial x_t}{\partial \theta}\mathrm{d}t.$

Untill Converge.

- The algorithm has a diffusion fine-tune step and a student model update step;
- $s_{p^{(t)}}(x_t, t)$ is teacher, $s_\phi(x_t, t)$ TA, and g_θ student;
- The only requests g_θ is that its generated sample is differentiable to θ;

Connection to DreamFusion:

- If the generator's output is a Dirac's Delta distribution with learnable parameters, i.e. $q(x_0) = \delta_{g(\theta)}(x_0)$ [a].
- Then the Diff-Instruct gradient formula becomes

$$\mathrm{Grad}(\theta) = \int_{t=0}^T w(t)\mathbb{E}_{\substack{x_0=g(\theta),\\ x_t|x_0\sim p_t(x_t|x_0)}}\big[\nabla_{x_t}\log p_t(x_t|x_0) - s_{p^{(t)}}(x_t)\big]\frac{\partial x_t}{\partial \theta}\mathrm{d}t.$$

- This shows that **DreamFusion is a special case of Diff-Instruct**.

[a] We switch the notation from $g_\theta(z)$ to $g(\theta)$ since under the assumptions the generator has no randomness.

Application 1: 1-step Diffusion Distillation:

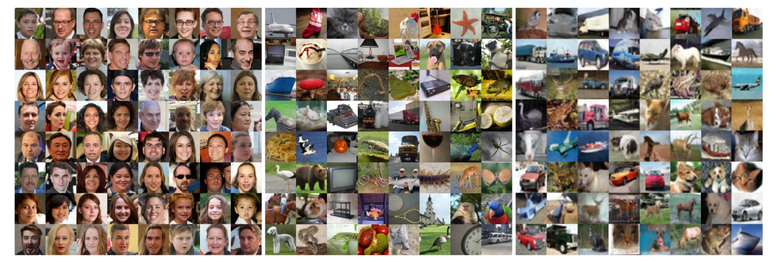

Figure 2: Generated samples from one-step generators that are distilled from pre-trained diffusion models on different datasets. *Left*: FFHQ-64 (unconditional); *Mid*: ImageNet-64 (conditional); *Right*: CIFAR-10 (unconditional).

Performance:

Table 1: Unconditional sample quality on CIFAR-10 through diffusion generations. *Methods that require synthetic data construction for distillation. †Methods that require real data for distillation.

METHOD	NFE (↓)	FID (↓)	IS (↑)
Multiple Steps (include Diffusion Distillation)			
DDPM [24]	1000	3.17	9.46
LSGM [70]	147	2.10	
PFGM [76]	110	2.35	9.68
EDM [34]	35	**1.97**	
DDIM [66]	50	4.67	
DDIM [66]	10	8.23	
DPM-solver-2 [43]	12	5.28	
DPM-solver-3 [43]	12	6.03	
3-DEIS [80]	10	4.17	
UniPC [82]	8	5.10	
UniPC [82]	5	23.22	
Denoise Diffusion GAN(T=2) [75]	2	4.08	**9.80**
PD [63]	2	5.58	9.05
CT [68]	2	5.83	8.85
CD† [68]	2	2.93	9.75
Single Step			
Denoise Diffusion GAN(T=1) [75]	1	14.6	8.93
KD* [44]	1	9.36	
TDPM [85]	1	8.91	8.65
1-ReFlow [42]	1	378	1.13
CT [68]	1	8.70	8.49
1-ReFlow (+distill)* [42]	1	6.18	9.08
2-ReFlow (+distill)* [42]	1	4.85	9.01
3-ReFlow (+distill)* [42]	1	5.21	8.79
PD [63]	1	8.34	8.69
CD-L2† [68]	1	7.90	
CD-LPIPS† [68]	1	**3.55**	9.48
Diff-Instruct	1	4.12	**9.89**

Table 2: Class-conditional sample quality on CIFAR-10 and ImageNet $\mathbf{64\times64}$ through diffusion generations. *Methods that require synthetic data construction for distillation. †Methods that require real data for distillation.

METHOD	NFE (↓)	FID (↓)
Multiple Steps (include Diffusion Distillation)		
EDM [34]	35	**1.79**
EDM-Heun [34]	20	2.54
EDM-Euler [34]	20	6.23
EDM-Heun [34]	10	15.56
Single Step		
EDM [34]	1	314.81
Diff-Instruct	1	**4.19**

Class-conditional ImageNet $\mathbf{64\times64}$. †Distillation techniques.

METHOD	NFE (↓)	FID (↓)
Multiple Steps		
ADM [13]	250	**2.07**
SN-DDIM [5]	100	17.53
EDM [34]	79	2.44
EDM-Heun[34]	10	17.25
GGDM [72]	25	18.4
CT [68]	2	11.1
PD† [63]	2	8.95
CD† [68]	2	4.70
Single Steps		
EDM[34]	1	154.78
PD† [63]	1	15.39
CT [68]	1	13.00
CD-L2† [68]	1	12.10
CD-LPIPS† [68]	1	6.20
Diff-Instruct	1	**4.24**

Application 2: Improving GAN models:

Table 3: Class-conditional sample quality on CIFAR10 through GAN models. † Models that we implemented.

METHOD	FID (↓)	IS (↑)
BigGAN [6]	14.73	9.22
BigGAN+Tune [6]	8.47	9.07 ± 0.13
MultiHinge [35]	6.40	9.58 ± 0.09
FQ-GAN [83]	5.59	8.48 ± 0.09
Stylegan2 [32]	6.96	9.53 ± 0.06
Stylegan2†	7.01	9.23 ± 0.07
Stylegan2† + DI	6.62	9.40 ± 0.06
Stylegan2+ADA [31]	3.49	$\mathbf{10.24} \pm 0.07$
Stylegan2+ADA+Tune [31]	2.42	10.14 ± 0.09
Stylegan2+ADA+Tune + DI	**2.27**	10.11 ± 0.10

Table 4: Unconditional sample quality on CIFAR10 through GAN models. † Models that we implemented.

METHOD	FID (↓)	IS (↑)
SNGAN [51]	21.70	8.22
ProGAN [18]	15.52	8.56 ± 0.06
AutoGAN [19]	12.42	8.55 ± 0.10
SNGAN+DGflow [1]	9.35	9.62
TransGAN [29]	9.02	9.26
StyleGAN2 [32]	8.32	9.21 ± 0.09
StyleGAN2†	8.21	9.09 ± 0.09
StyleGAN2† + DI	7.56	9.16 ± 0.09
StyleGAN2+ADA [31]	5.33	10.02 ± 0.07
StyleGAN2+ADA+Tune [31]	2.92	9.83 ± 0.04
StyleGAN2+ADA+Tune + DI	**2.71**	$\mathbf{9.86} \pm 0.04$

NeurIPS 2023 - New Orleans, LA