HarmoniCa: Harmonizing Training and Inference for Better Feature Caching in Diffusion Transformer Acceleration

Yushi Huang[1,2*], Zining Wang[2,3*], Ruihao Gong[2,3+], Jing Liu[4], Xinjie Zhang[1], Jinyang Guo[3], Xianglong Liu[3], Jun Zhang[1+]

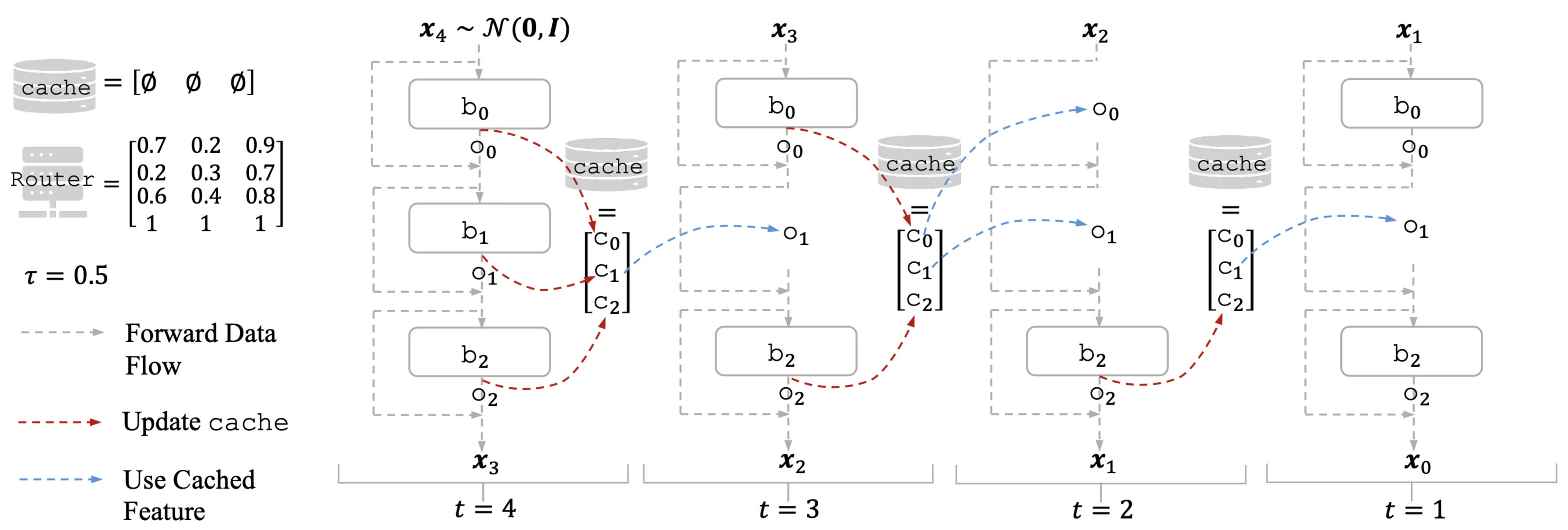
[1]Hong Kong University of Science and Technology [2]SenseTime [3]Beihang University [4]Monash University

* : Equal contribution
+: Corresponding authors

ICML International Conference On Machine Learning

Feature Caching

High similarity between features across denoising steps enables caching and reusing these features to avoid redundant computations.

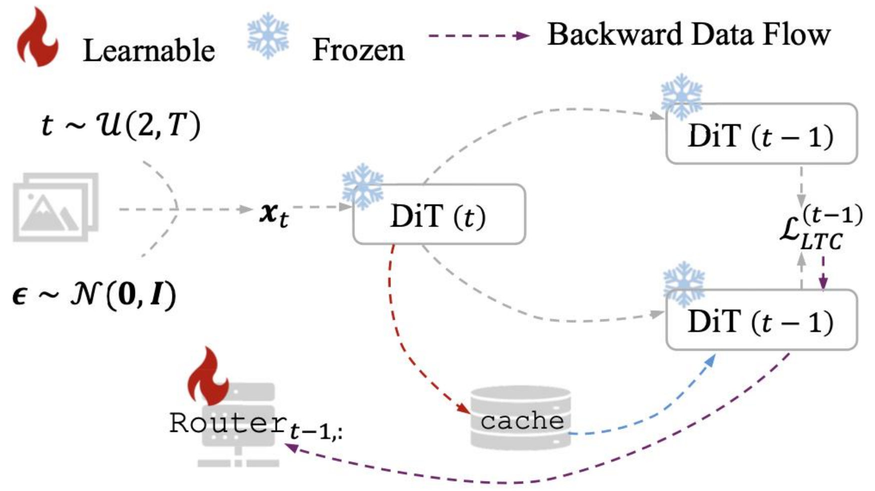


Q: How to determine **when** and **where** to cache and reuse?
A: **Learn** an optimal `Router` (superior than heuristic ways).

Discrepancy between Training & Inference

1. Prior timestep disregard

Infer: $\boldsymbol{x}_1$ is pertubed by $\circ_0$ and $\circ_1$; Contents of the current `cache` are shaped by reuse and update.
Train: $\boldsymbol{x}_{t-1}$ is accurate; contents of the current `cache` are only determined by DiT (t).



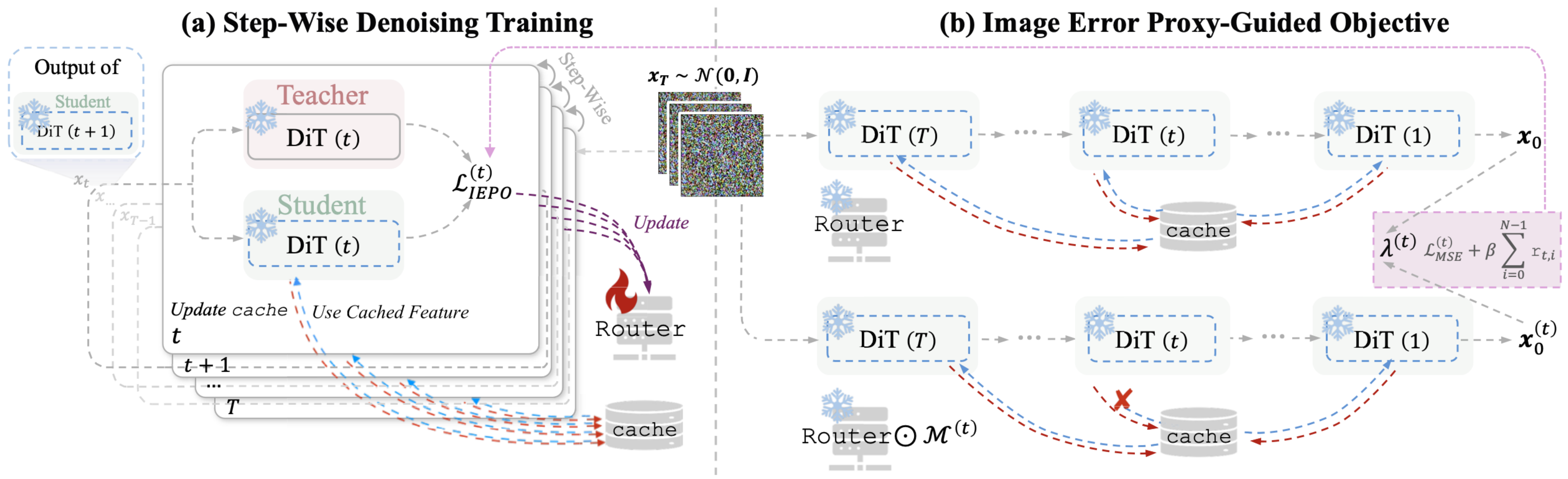
2. Objective mismatch

Infer: Generate high quality $\boldsymbol{x}_0$.
Train: Solely align the predicted noise at each denoising step.

$$\mathcal{L}_{LTC}^{(t)} = \mathcal{L}_{MSE}^{(t)} + \beta \sum_{i=0}^{N-1} \mathrm{r}_{t,i}$$

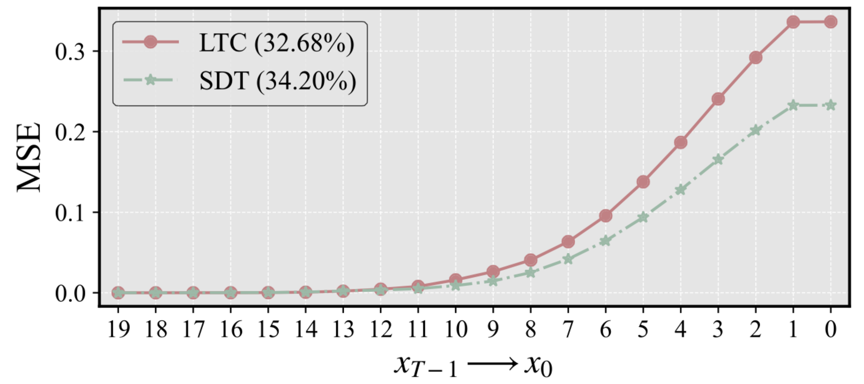
Paper **Code**

Harmonizing Training & Inference



1. Step-wise denoising training

This mimics the multi-timestep inference stage, which integrates the impact of prior timesteps at t. Moreover, this strategy requires **zero training images**.



The Mean Square Error (MSE) of $\mathbf{x}_t$ for DiT-XL/2 256 ×256 induced by different feature caching methods.

2. Image error proxy-guided objective

The straightforward solution (*i.e.,* direct optimize $\boldsymbol{x}_0$) incurs $10\times$ memory and $5\times$ time consumption for training. Thus, we propose a proxy $\boldsymbol{\lambda}^{(t)}$ to consider the trade-off between the error of $\boldsymbol{x}_0$ and the `cache` usage at a certain denoising step.

$$\mathcal{L}_{IEPO}^{(t)} = \boldsymbol{\lambda}^{(t)} \mathcal{L}_{MSE}^{(t)} + \beta \sum_{i=0}^{N-1} \mathrm{r}_{t,i}$$

$$\mathcal{M}^{(t)} = \begin{cases} 1, j \neq t \\ \frac{1}{\mathrm{r}_{j,k}}, j = t \end{cases} \Rightarrow \boldsymbol{\lambda}^{(t)} = \left\| x_0 - x_0^{(t)} \right\|_F^2$$

(a) DiT-XL/2 (b) SDT+$\mathcal{L}_{LTC}^{(t)}$ (1.40×) (c) HarmoniCa (1.44×)

Random samples for DiT-XL/2 256 ×256 w/ and w/o feature caching ($T = 20$). We mark the speedup ratio in the brackets. HarmoniCa means our SDT equipped with IEPO.

Experiments

Text-to-image generation.

Method	T	CLIP↑	FID↓	sFID↓	CUR(%)↑	Latency(s)↓
PixArt-α 256 × 256 (cfg = 4.5)						
DPM-Solver++ (Lu et al., 2022b)	20	30.96	27.68	36.39	-	0.553
DPM-Solver++ (Lu et al., 2022b)	15	30.77	31.68	38.92	-	$0.418_{(1.32\times)}$
FORA (Selvaraju et al., 2024)	20	31.10	27.42	37.98	50.00	$0.364_{(1.52\times)}$
HarmoniCa	20	**31.13**	**26.33**	**37.85**	**56.01**	$\mathbf{0.346}_{(1.60\times)}$
IDDPM (Nichol & Dhariwal, 2021)	100	31.25	24.15	33.65	-	2.572
IDDPM (Nichol & Dhariwal, 2021)	75	31.25	24.17	33.73	-	$1.868_{(1.37\times)}$
FORA (Selvaraju et al., 2024)	100	31.25	25.16	33.62	50.00	$1.558_{(1.65\times)}$
HarmoniCa	100	**31.17**	**23.73**	**32.23**	**53.24**	$\mathbf{1.523}_{(1.69\times)}$
SA-Solver (Xue et al., 2024)	25	31.31	26.78	38.35	-	0.891
SA-Solver (Xue et al., 2024)	20	31.23	27.45	39.01	-	$0.665_{(1.34\times)}$
HarmoniCa	25	**31.27**	**27.07**	**38.62**	**54.19**	$\mathbf{0.561}_{(1.59\times)}$
PixArt-α 512 × 512 (cfg = 4.5)						
DPM-Solver++ (Lu et al., 2022b)	20	31.30	23.96	40.34	-	1.759
DPM-Solver++ (Lu et al., 2022b)	15	**31.29**	25.12	40.37	-	$1.291_{(1.36\times)}$
HarmoniCa	20	**31.29**	**24.81**	**40.18**	**54.64**	$\mathbf{1.072}_{(1.64\times)}$
SA-Solver (Xue et al., 2024)	25	31.23	25.43	39.84	-	2.263
SA-Solver (Xue et al., 2024)	20	31.19	25.85	40.08	-	$1.738_{(1.30\times)}$
HarmoniCa	25	**31.20**	**25.74**	**39.99**	**54.24**	$\mathbf{1.406}_{(1.61\times)}$
PixArt-α 1024 × 1024 (cfg = 4.5)						
DPM-Solver++ (Lu et al., 2022b)	20	31.10	25.01	37.80	-	9.470
DPM-Solver++ (Lu et al., 2022b)	15	31.07	25.77	42.50	-	$7.141_{(1.32\times)}$
HarmoniCa	20	**31.09**	**23.02**	**36.24**	**55.06**	$\mathbf{5.786}_{(1.63\times)}$
SA-Solver (Xue et al., 2024)	25	31.05	23.65	38.12	-	11.931
SA-Solver (Xue et al., 2024)	20	31.02	23.88	39.41	-	$9.209_{(1.30\times)}$
Harmonica	25	**31.07**	**23.77**	**38.93**	**53.98**	$\mathbf{7.551}_{(1.58\times)}$

Comparison with quantization and pruning.

Method	T	IS↑	FID↓	sFID↓	Latency(s)↓	Latency(s)↓*
DDIM (Zhang et al., 2022)	20	224.37	3.52	4.96	0.658	1.217
EfficientDM (He et al., 2024)	20	172.70	6.10	**4.55**	$0.591_{(1.11\times)}$	$0.842_{(1.45\times)}$
PTQ4DiT (Wu et al., 2024)	20	17.06	71.82	23.16	$0.577_{(1.14\times)}$	$0.839_{(1.45\times)}$
Diff-Pruning (Fang et al., 2023)	20	168.10	8.22	6.20	$0.458_{(1.44\times)}$	$\mathbf{0.813}_{(1.50\times)}$
HarmoniCa	20	**206.57**	**4.88**	5.91	$\mathbf{0.456}_{(1.44\times)}$	$0.815_{(1.49\times)}$

Combination with quantization.

Method	IS↑/CLIP↑	FID↓	sFID↓	CUR(%)↑	Latency(s)↓	#Size(GB)↓
DiT-XL/2 256 × 256 (cfg = 1.5)						
EfficientDM (He et al., 2024)	172.70	6.10	4.55	-	$0.591_{(1.11\times)}$	$0.64_{(3.93\times)}$
w/ HarmoniCa ($\beta = 4e^{-8}$)	168.16	6.48	4.32	26.25	$0.473_{(1.40\times)}$	$0.64_{(3.93\times)}$
PixArt-α 256 × 256 (cfg = 4.5)						
EfficientDM (He et al., 2024)	30.09	34.84	30.34	-	$0.469_{(1.18\times)}$	$0.59_{(1.98\times)}$
w/ HarmoniCa	30.15	34.96	30.55	53.34	$0.299_{(1.85\times)}$	$0.59_{(1.98\times)}$
PixArt-α 512 × 512 (cfg = 4.5)						
EfficientDM (He et al., 2024)	30.71	25.82	41.64	-	$0.461_{(1.20\times)}$	$0.59_{(1.98\times)}$
w/ HarmoniCa	30.75	26.15	41.99	53.11	$0.281_{(1.97\times)}$	$0.59_{(1.98\times)}$

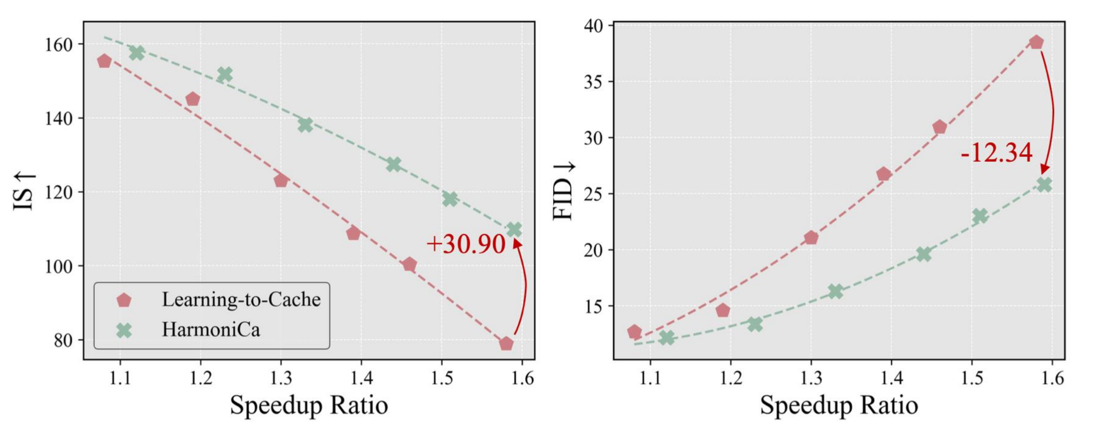
(Left) Comparison with addition caching methods on U-ViT; (Right) Training costs for DiT-XL/2 256 ×256.

Method	T	FID↓	Latency(s)↓
DPM-Solver (Lu et al., 2022a)	20	2.57	7.60
Faster Diffusion (Li et al., 2023a)	20	2.82	$5.95_{(1.28\times)}$
DeepCache (Ma et al., 2024b)	20	2.70	$4.68_{(1.62\times)}$
HarmoniCa	20	**2.61**	$\mathbf{4.60}_{(1.65\times)}$

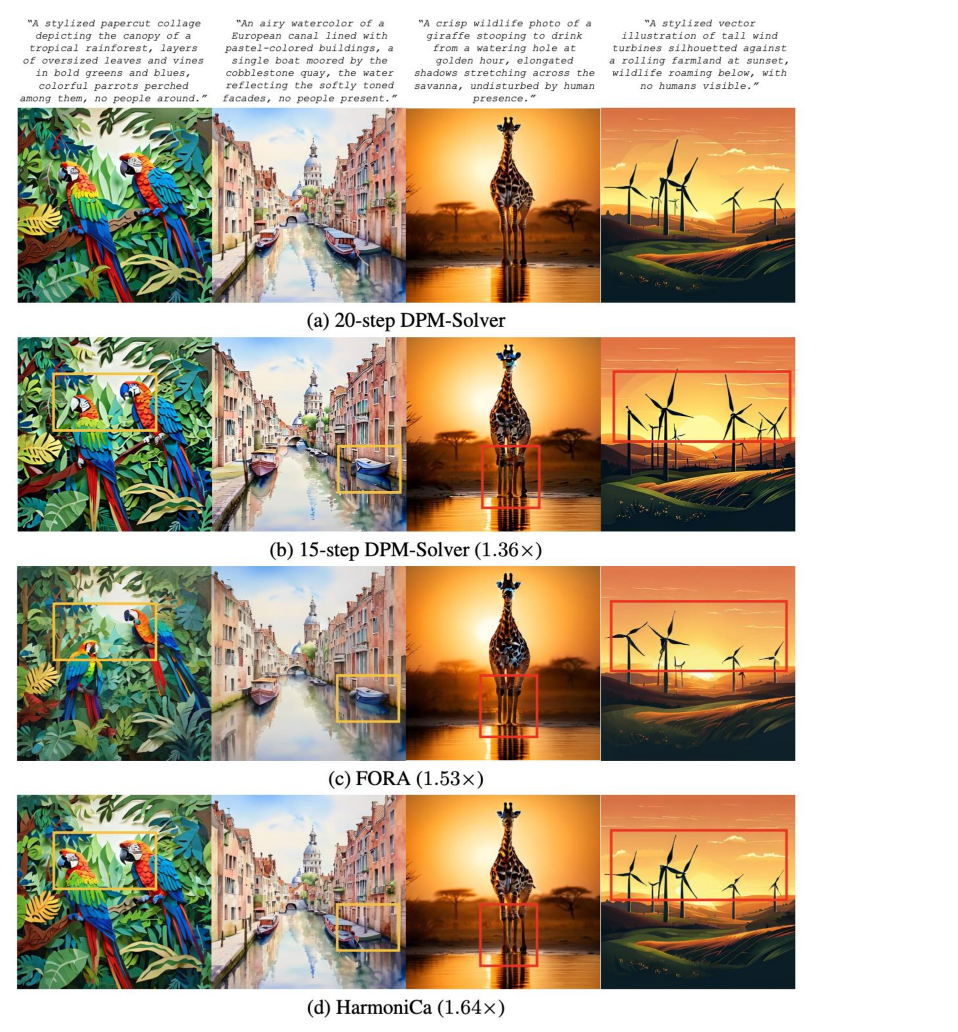
Method	#Images	Time(h)	Memory(GB/GPU)
Learning-to-Cache	1.22M	2.15	33.33
SDT+$\mathcal{L}_{LTC}^{(t)}$	0	1.47	33.28
HarmoniCa	0	1.63	33.28

Class-conditional generation.

Method	T	IS↑	FID↓	sFID↓	Prec.↑	Recall↑	CUR(%)↑	Latency(s)↓
DiT-XL/2 256 × 256 (cfg = 1.5)								
DDIM (Song et al., 2020a)	50	240.37	2.27	4.25	80.25	59.77	-	1.767
DDIM (Song et al., 2020a)	39	237.84	2.37	4.32	80.22	59.31	-	$1.379_{(1.28\times)}$
Learning-to-Cache (Ma et al., 2024a)	50	233.26	2.62	4.50	79.40	59.15	23.39	$1.419_{(1.25\times)}$
HarmoniCa	50	**238.74**	**2.36**	**4.24**	**80.57**	**59.68**	**23.68**	$\mathbf{1.361}_{(1.30\times)}$
DDIM (Song et al., 2020a)	20	224.37	3.52	4.96	78.47	58.33	-	0.658
DDIM (Song et al., 2020a)	14	201.83	5.77	6.61	75.14	55.08	-	$0.466_{(1.41\times)}$
Learning-to-Cache (Ma et al., 2024a)	20	201.37	5.34	6.36	75.04	56.09	35.60	$0.468_{(1.41\times)}$
HarmoniCa	20	**206.57**	**4.88**	**5.91**	**75.20**	**58.74**	**37.50**	$\mathbf{0.456}_{(1.44\times)}$
DDIM (Song et al., 2020a)	10	159.93	12.16	11.31	67.10	52.27	-	0.332
DDIM (Song et al., 2020a)	9	140.37	16.54	14.44	62.63	50.08	-	$0.299_{(1.11\times)}$
Learning-to-Cache (Ma et al., 2024a)	10	145.09	14.59	11.58	64.03	52.06	19.11	$0.279_{(1.19\times)}$
HarmoniCa	10	**151.83**	**13.35**	**11.13**	**65.22**	**52.18**	**22.86**	$\mathbf{0.270}_{(1.23\times)}$
DiT-XL/2 512 × 512 (cfg = 1.5)								
DDIM (Song et al., 2020a)	20	184.47	5.10	5.79	81.77	54.50	-	3.356
DDIM (Song et al., 2020a)	16	173.31	6.47	6.67	81.10	51.30	-	$2.688_{(1.25\times)}$
Learning-to-Cache (Ma et al., 2024a)	20	178.11	6.24	7.01	81.21	53.30	23.57	$2.633_{(1.28\times)}$
HarmoniCa	20	**179.84**	**5.72**	**6.61**	**81.33**	**55.80**	**25.98**	$\mathbf{2.574}_{(1.30\times)}$



Comparison with Learning-to-Cache with the increase of the speedup ratio.



Visualization results.