Few-shot Learner Parameterization by Diffusion Time-steps

Appendix

Abbreviation/Symbol	Meaning
	Abbreviations
FSL	Few-Shot Learning
DM	Diffusion Model
SD	Stable Diffusion
LoRA	Low-Rank Adaptation
	Symbols in Theory
$\mathbf{c}, \mathcal{C}$	Class attribute, the set of all class attributes
$\mathbf{e}, \mathcal{E}$	Environmental attribute, the set of all environmental attributes
Φ	Generator mapping from $\mathcal{C} \times \mathcal{E}$ to the image space
$\mathrm{erf}(z)$	$\frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$
τ	An attribute loss degree threshold
$\mathrm{Err}(\cdot)$	Attribute loss degree
$t(\mathbf{c}), t(\mathbf{e})$	Time-step when $\mathbf{c}$ or $\mathbf{e}$ is lost with at least degree τ
	Symbols in Approach
K-way-N-shot	FSL task on K classes with N training images in each one
$\mathbf{x}$	Image
$\mathbf{x}_t$	Image with injected noise at time-step t
c	Class c
y, y_c	A textual prompt, prompt describing class c
$\beta_1, \ldots, \beta_T$	Variance schedule
$\alpha_t,\ \bar{\alpha}_t$	$1 - \beta_t,\ \prod_{s=1}^{t} \alpha_s$
$q(\mathbf{x}_t \vert \mathbf{x}_0)$	Noisy sample distribution (from DM forward process)
T	Total time-steps
$d(\cdot)$	DM denoising network
$d(\cdot; \theta_c)$	DM denoising network injected by LoRA specific to class c
[V]	A rare token identifier
$\mathcal{L}_t$	DM reconstruction loss at time-step t
w_t	Standard, pre-defined weight for reconstruction loss at t
r_t	Ratio of class attribute loss degree over that of all attribute losses
δ^*	Average pixel-level changes when only altering class attribute
γ_t	Weight term from Eq. (5) used to compute r_t

Table A1. List of abbreviations and symbols.

This appendix is organized as follows:

- Section A1 derives the closed-form of Eq. (5) and provides the full proof to our Theorem.
- Section A2 provides additional implementation details, including a detailed DM formulation, TiF learner training and inference algorithm as well as details of the attention map in Figure 2 and our text prompts.
- Section A3 shows additional results, including FSL accuracy on additional tasks, ablation on Stable Diffusion (SD) version and LoRA insertion location. We also discuss show results of learning a single LoRA for all classes, instead of learning class-specific LoRAs.

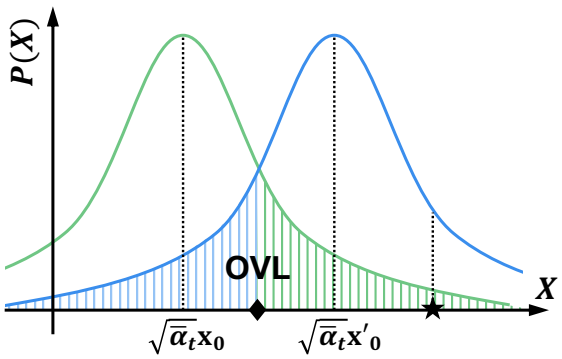

Figure A1. The PDFs of $q(\mathbf{x}_t|\mathbf{x}_0)$ (green) and $q(\mathbf{x}'_t|\mathbf{x}'_0)$ (blue). Without loss of generality, we consider the 1-D case of $x'_0 > x_0$. Their means are computed from Eq. (3).

A1. Proof of Theory

Theorem. *1) For each* $\mathbf{c} \in \mathcal{C}$*, there exists a smallest time-step* $t(\mathbf{c})$*, such that* $\mathbf{c}$ *is lost with at least degree* τ *at each* $t \in \{t(\mathbf{c}), \dots, T\}$*. This also holds for each* $\mathbf{e} \in \mathcal{E}$*. 2)* $\exists\{\beta_i\}_{i=1}^T$ *such that* $t(\mathbf{e}) > t(\mathbf{c})$ *whenever* $\|\Phi(\mathbf{c}', \mathbf{e}) - \Phi(\mathbf{c}', \mathbf{e}')\|$ *is first-order stochastic dominant over* $\|\Phi(\mathbf{c}, \mathbf{e}') - \Phi(\mathbf{c}', \mathbf{e}')\|$ *with* $\mathbf{c}' \sim \mathcal{C}, \mathbf{e}' \sim \mathcal{E}$ *uniformly.*

Proof. We start by showing $\mathrm{Err}(\mathbf{x}_0, \mathbf{x}'_0, t) = \frac{1}{2}\mathrm{OVL}\left(q(\mathbf{x}_t|\mathbf{x}_0), q(\mathbf{x}'_t|\mathbf{x}'_0)\right)$. Without loss of generality, we show a 1-D sample space $\mathcal{X}$ in Figure A1. The minimum Err_θ is obtained when given each noisy sample $\mathbf{x}$, DM reconstructs towards $\mathbf{x}_0$ if $q(\mathbf{x}|\mathbf{x}_0) > q(\mathbf{x}|\mathbf{x}'_0)$ and vice versa for $\mathbf{x}'_0$, *e.g.*, reconstructing $\star$ as $\mathbf{x}'_0$. However, this maximum likelihood estimation fails when a noisy sample is drawn from $q(\mathbf{x}_t|\mathbf{x}_0)$ (green PDF), but with a value larger than the intersection point of the two PDFs (♦), and similar arguments go for $q(\mathbf{x}'_t|\mathbf{x}'_0)$ (blue PDF). The error rate caused by the two failure cases corresponds to the green shaded area and blue one, respectively, leading to an average Err_θ of $\frac{1}{2}$ of the OVL.

To compute the OVL, it is trivial in the 1-D case by leveraging the Cumulative Distribution Function (CDF) of Gaussian distribution. Given that the two distributions have equal variance from Eq. (3), the intersection point is given by $\frac{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{\bar{\alpha}_t}\mathbf{x}'_0}{2}$. For a Gaussian distribution $\mathcal{N}(\mu, \sigma^2)$, its CDF is given by $\frac{1}{2}\left[1 + \mathrm{erf}(\frac{x-\mu}{\sqrt{2}\sigma})\right]$. Combining two results, one can easily show that the blue shaded area, corresponding to half of the OVL, or $\mathrm{Err}(x_0, y_0, t)$, is given by:

$$\begin{aligned}\mathrm{Err}(x_0, y_0, t) &= \frac{1}{2}\mathrm{OVL}\left(q(x_t|x_0), q(y_t|y_0)\right) \\ &= \frac{1}{2}\left[1 - \mathrm{erf}\left(\frac{\sqrt{\bar{\alpha}_t}(y_0 - x_0)}{2\sqrt{2(1-\bar{\alpha}_t)}}\right)\right].\end{aligned} \tag{A1}$$

To generalize the results to multi-variate Gaussian distributions, we use the results in [5], which shows that by projecting the data to Fisher's linear discriminant axis, the OVL defined on the discriminant densities is equal to that defined on the multivariate densities. Specifically, the mean of the discriminant densities are given by

$$\mu_0 = \sqrt{\bar{\alpha}_t}(\mathbf{x}'_0 - \mathbf{x}_0)^\top \Sigma^{-1}\mathbf{x}_0, \;\; \mu_1 = \sqrt{\bar{\alpha}_t}(\mathbf{x}'_0 - \mathbf{x}_0)^\top \Sigma^{-1}\mathbf{x}'_0, \tag{A2}$$

where $\Sigma = \beta_t \mathbf{I}$. The common variance of the discriminant densities is given by $\sqrt{\bar{\alpha}_t}(\mathbf{x}'_0 - \mathbf{x}_0)^\top \Sigma^{-1}(\mathbf{x}'_0 - \mathbf{x}_0)$. Following the calculation steps to compute OVL for the 1-D case, one can show that for both 1-D and multi-variate case, we have

$$\begin{aligned}\mathrm{Err}(\mathbf{x}_0, \mathbf{x}'_0, t) &= \frac{1}{2}\mathrm{OVL}\left(q(\mathbf{x}_t|\mathbf{x}_0), q(\mathbf{x}'_t|\mathbf{x}'_0)\right) \\ &= \frac{1}{2}\left[1 - \mathrm{erf}\left(\frac{\|\sqrt{\bar{\alpha}_t}(\mathbf{x}'_0 - \mathbf{x}_0)\|}{2\sqrt{2(1-\bar{\alpha}_t)}}\right)\right].\end{aligned} \tag{A3}$$

As $\bar{\alpha}_t$ decreases with an increasing t from Eq. (3), and the error function $\mathrm{erf}(\cdot)$ is strictly increasing, $\mathrm{Err}(\mathbf{x}_0, \mathbf{x}'_0, t)$ is strictly increasing in t given any $\mathbf{x}_0, \mathbf{x}'_0$. Hence $\mathbb{E}_{(\mathbf{c}',\mathbf{e})\in\mathcal{C}\times\mathcal{E}}\left[\mathrm{Err}\left(\Phi(\mathbf{c}, \mathbf{e}), \Phi(\mathbf{c}', \mathbf{e}), t\right)\right] > \mathbb{E}_{(\mathbf{c}',\mathbf{e})\in\mathcal{C}\times\mathcal{E}}\left[\mathrm{Err}\left(\Phi(\mathbf{c}, \mathbf{e}), \Phi(\mathbf{c}', \mathbf{e}), t(\mathbf{c})\right)\right]$ for every $t \geq t(\mathbf{c})$, which completes the proof of the first part of the Theorem.

To prove the second part of the Theorem, let F_1 and F_2 denote the cumulative distribution function of $\delta_\mathbf{e} = \|\Phi(\mathbf{c}', \mathbf{e}) - \Phi(\mathbf{c}', \mathbf{e}')\|$ and $\delta_\mathbf{c} = \|\Phi(\mathbf{c}, \mathbf{e}') - \Phi(\mathbf{c}', \mathbf{e}')\|$, respectively (under uniform sampling of $\mathbf{c}', \mathbf{e}'$), and let f_1, f_2 be the probability density function of F_1, F_2, respectively. Then we have:

$$\mathop{\mathbb{E}}_{(\mathbf{c}',\mathbf{e}')\in\mathcal{C}\times\mathcal{E}} \mathrm{erf}(\gamma_t\delta_\mathbf{e}) - \mathop{\mathbb{E}}_{(\mathbf{c}',\mathbf{e}')\in\mathcal{C}\times\mathcal{E}} \mathrm{erf}(\gamma_t\delta_\mathbf{c}) \tag{A4}$$

$$= \int_0^\infty \mathrm{erf}(\gamma_t\delta) f_1(\delta) d\delta - \int_0^\infty \mathrm{erf}(\gamma_t\delta) f_2(\delta) d\delta \tag{A5}$$

$$= \int_0^\infty \gamma_t \mathrm{erf}'(\gamma_t\delta)\left[F_2(\delta) - F_1(\delta)\right] d\delta > 0, \tag{A6}$$

In Eq. (A4), $\gamma_t = \sqrt{\bar{\alpha}_t}/\sqrt{8(1-\bar{\alpha}_t)}$ is the constant in Eq. (A1). Eq. (A5) is from the definition of expectation and the uniform sampling of $\mathbf{c}', \mathbf{e}'$ in (Y). Eq. (A6) is derived from integrating by parts. Eq. (A6) is positive as I) γ_t is positive by definition for all t; II) $\mathrm{erf}(\cdot)$ is strictly increasing, hence the derivative $\mathrm{erf}'(\cdot) > 0$; III) By definition of first-order stochastic domination, $F_2(\delta) - F_1(\delta) \geq 0$ for *all* δ and $F_2(\delta) - F_1(\delta) > 0$ for *some* δ. As Eq. (A4) is larger than 0, we can arrive at $\mathbb{E}_{(\mathbf{c}',\mathbf{e}')\in\mathcal{C}\times\mathcal{E}}\left[\mathrm{Err}\left(\Phi(\mathbf{c}', \mathbf{e}), \Phi(\mathbf{c}', \mathbf{e}'), t\right)\right] < \mathbb{E}_{(\mathbf{c}',\mathbf{e}')\in\mathcal{C}\times\mathcal{E}}\left[\mathrm{Err}\left(\Phi(\mathbf{c}, \mathbf{e}'), \Phi(\mathbf{c}', \mathbf{e}'), t\right)\right]$ by definition of $\mathrm{Err}(\cdot)$ in Eq. (A1).

Next, we show that $\exists T, \{\beta_i\}_{i=1}^T$ such that the aforementioned inequality implies $t(\mathbf{e}) > t(\mathbf{c})$. The variance schedule should satisfy: $\exists t$ such that

$\mathbb{E}_{(\mathbf{c}',\mathbf{e}')\in\mathcal{C}\times\mathcal{E}}\left[\mathrm{Err}\left(\Phi(\mathbf{c}',\mathbf{e}),\Phi(\mathbf{c}',\mathbf{e}'),t\right)\right] < \tau \leq \mathbb{E}_{(\mathbf{c}',\mathbf{e}')\in\mathcal{C}\times\mathcal{E}}\left[\mathrm{Err}\left(\Phi(\mathbf{c},\mathbf{e}'),\Phi(\mathbf{c}',\mathbf{e}'),t\right)\right]$ (*i.e.*, $\mathbf{e}$ not lost while $\mathbf{c}$ lost). This rules out the corner case where $t(\mathbf{e}) = t(\mathbf{c})$, *e.g.*, by setting $\beta_1 \approx 1$, $\mathbf{x}_1$ is already closed to pure noise and loses all attributes immediately. Note that this is unlikely in practice as noise is gradually added.

A2. Implementation Details

A2.1. DM Formulation

Reverse Process. DM training corresponds to a learned Gaussian transition $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ parameterized by θ, starting at $p(\mathbf{x}_T) := \mathcal{N}(\mathbf{x}_T;\mathbf{0},\mathbf{I})$. Each $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$ is computed in two steps: 1) Reconstruct $\mathbf{x}_0$ from $\mathbf{x}_t$ with $u_\theta(\mathbf{x}_t,t)$, where u_θ is a learnable U-Net [8]. 2) Compute $q\left(\mathbf{x}_{t-1}|\mathbf{x}_t,u_\theta(\mathbf{x}_t,t)\right)$, which has a closed-form solution. It is given by $\mathcal{N}(\mathbf{x}_{t-1}|\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0),\tilde{\beta}_t\mathbf{I})$, where

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t,\mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t, \quad \text{(A7)}$$

where $\tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$.

Equivalent Formulation. The simplified objective in DDPM [2] is given by:

$$\mathcal{L}_{DM} = \mathop{\mathbb{E}}_{t,\mathbf{x}_0,\epsilon} \|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, y, t)\|^2, \quad \text{(A8)}$$

where ϵ_θ is a θ-parameterized U-Net [8] to predict the added noise. From Eq. (3), we have

$$\epsilon = \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}\mathbf{x}_0}{\sqrt{1-\bar{\alpha}_t}}, \ \epsilon_\theta(\mathbf{x}_t,y,t) = \frac{\mathbf{x}_t - \sqrt{\bar{\alpha}_t}d(\mathbf{x}_t,y,t)}{\sqrt{1-\bar{\alpha}_t}}, \quad \text{(A9)}$$

Taking Eq. A9 into Eq. A8 yields Eq. (4) with the corresponding $w_t = \bar{\alpha}_t/(1-\bar{\alpha}_t)$.

A2.2. TiF Algorithm

Our training is summarized in Algorithm 1. In particular, we train class-specific LoRA matrices and use a fixed text prompt y. We tried training one set of LoRA matrices for all classes with class-specific prompts y_c. We include the results in Section A3, where it performs poorly. Note that our current approach also enjoys the added benefit of naturally supporting class-incremental learning, as the models for existing classes do not need to be retrained when adding new classes.

We follow the evaluation pipeline in Diffusion Classifier [4], which enables us to fairly compare with their results. Specifically, instead of sampling t,ϵ for each class to compute Eq. (4), the pipeline divides the evaluation to several stages, where each stage progressively removes unlikely classes from all classes $\{1,\dots,K\}$, until a single class is left as prediction after the last stage. Specifically, one need to specify a sequence S denoting the number of classes left after each stage, and a sequence M denoting the size of a time-step subset used to compute Eq. (7). For example, if $S = [20,10,5,1], M = [20,50,100,330]$ and $K = 100$, in the first stage, we sample a single ϵ for each time-step in $\{0,50,100,\dots,950\}$ (20 time-steps) to evaluate Eq. (7) for all 100 classes, and select 20 classes after the first stage. We summarize the pipeline in Algorithm 2. For FGVCAircraft, we used $S = [20,10,5,1], M = [20,50,100,330]$. For ISIC2019, we used $S = [4,2,1], M = [128,256,116]$. For VeRi-776 and DukeMTMC-reID, we used $S = [50,20,5,1], M = [32,128,256,84]$. Note that we designed these values based on K and did not perform any search. We leave improvements to the evaluation pipeline as future work.

Algorithm 1: TiF Learner Training

Input : Few-shot training set $\mathcal{D}$, pre-trained stable diffusion d, fixed prompt y.

for $i = 1,\dots,K$ **do**
  Select images in class i as $\mathcal{D}_i = \{\mathbf{x}|c=i\}$;
  Randomly initialize θ_i;
  while *not converged* **do**
    Sample $\mathbf{x}_0 \sim \mathcal{D}_i$;
    Sample $t \in \{1,\dots,T\}$, $\epsilon \sim \mathcal{N}(\mathbf{0},\mathbf{I})$;
    $\hat{\mathbf{x}}_0 \leftarrow d(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, y, t \mid \theta_i)$;
    Update θ_i to minimize $w_t\|\mathbf{x}_0 - \hat{\mathbf{x}}_0\|^2$
return *Optimized* $\{\theta_i\}_{i=1}^K$.

Algorithm 2: TiF Learner Inference

Input : Optimized $\{\theta_i\}_{i=1}^K$, pre-trained stable diffusion d, y, test image $\mathbf{x}$, number of stages s, number of selected classes in each stage S such that $S[s]=1$, number of used time-steps in each stage M.

$\mathcal{S} \leftarrow \{1,\dots,K\}$;
Compute r_t for $t \in \{1,\dots,T\}$ with Eq. (9);
Set $e_c \leftarrow 0$ for each $c \in \{1,\dots,K\}$;
for $i = 1,\dots,s$ **do**
  for $c \in \mathcal{S}$ **do**
    $\mathcal{T} \leftarrow \mathrm{arange}(0,T,T/M[s])$;
    for $t \in \mathcal{T}$ **do**
      Sample $\mathbf{x}_t \sim q(\mathbf{x}_t|\mathbf{x})$;
      $e \leftarrow -r_t\|\mathbf{x} - d(\mathbf{x}_t,y,t \mid \theta_c)\|^2$;
      $e_c \leftarrow e_c + e$;
  Select top-$S[i]$ classes c in $\mathcal{S}$ with largest e_c;
  $\mathcal{S} \leftarrow$ selected top-$S[i]$ classes;
$\hat{c} \leftarrow \mathcal{S}$;
return *Prediction* $\hat{c} \in \{1,\dots,K\}$.

	Method	FGVCAircraft					ISIC2019				
		1	2	4	8	16	1	2	4	8	16
CLIP	Zero-Shot [7]			24.9					12.5		
	CoOp [12]	25.3	27.7	35.7	35.1	39.9	13.0	12.2	19.9	22.4	23.3
	Co-CoOp [11]	27.4	28.9	33.5	35.9	37.9	10.0	12.6	15.1	16.1	17.7
	MaPLe* [3]	27.7	31.0	33.7	40.8	45.2	11.2	13.1	16.7	18.9	23.2
OpenCLIP	Zero-Shot [7]			42.3					16.9		
	Linear-probe [7]	17.9	33.3	44.2	53.3	59.9	10.3	12.5	17.9	20.2	21.8
	Tip-Adapter [9]	46.9	49.2	53.0	57.4	59.4	**20.3**	22.1	26.6	31.7	35.6
	Tip-Adapter-F [9]	49.8	52.7	56.9	61.7	67.6	19.2	22.6	27.5	32.6	37.5
	CaFo* [10]	**50.2**	53.8	58.7	62.9	67.0	-	-	-	-	-
DM	Zero-Shot [4]			24.3					11.7		
	TiF learner	48.4	**54.4**	**66.2**	**72.3**	**79.3**	**20.3**	**23.2**	**31.0**	**34.8**	**40.1**

Table A2. N-shot accuracies on FGVCAircraft and ISIC2019 with seed 2. Supplementary to Table 1 (seed 1).

	Method	FGVCAircraft					ISIC2019				
		1	2	4	8	16	1	2	4	8	16
CLIP	Zero-Shot [7]			24.9					12.5		
	CoOp [12]	24.1	29.7	32.4	37.7	40.5	15.9	20.9	18.2	20.8	26.4
	Co-CoOp [11]	29.2	29.1	31.5	37.4	40.7	14.0	15.6	16.1	18.6	23.6
	MaPLe* [3]	22.5	28.2	33.6	39.4	46.1	12.6	14.0	18.4	21.2	27.1
OpenCLIP	Zero-Shot [7]			42.3					16.9		
	Linear-probe [7]	21.1	35.8	43.6	55.6	58.6	12.8	16.3	17.9	20.6	22.8
	Tip-Adapter [9]	47.0	50.6	53.0	57.0	60.1	23.0	**28.3**	28.8	31.7	35.6
	Tip-Adapter-F [9]	50.5	52.2	55.6	62.4	67.0	**23.2**	26.2	24.3	32.5	38.2
	CaFo* [10]	**51.4**	53.4	58.0	62.7	65.9	-	-	-	-	-
DM	Zero-Shot [4]			24.3					11.7		
	TiF learner	47.7	**57.9**	**64.3**	**72.5**	**79.5**	18.5	25.5	**30.3**	**33.2**	**41.6**

Table A3. N-shot accuracies on FGVCAircraft and ISIC2019 with seed 3. Supplementary to Table 1 (seed 1).

	Method	DukeMTMC-reID					VeRi-776				
		1	2	4	8	16	1	2	4	8	16
CLIP	CoOp [12]	8.2	10.4	17.2	29.6	33.7	11.4	13.9	18.1	30.3	34.1
	Co-CoOp [11]	5.4	9.8	20.2	31.1	42.4	28.9	31.0	32.3	36.9	38.0
	MaPLe [3]	11.4	31.6	41.2	54.5	61.1	33.2	41.5	42.7	56.5	66.8
OC	Linear-probe [7]	10.5	15.8	42.2	50.7	63.1	11.7	29.3	50.9	60.2	68.6
	Tip-Adapter [9]	**31.1**	37.0	45.9	60.1	68.6	35.3	43.8	61.3	72.2	79.5
	Tip-Adapter-F [9]	29.8	33.5	54.6	74.1	85.7	35.1	46.8	62.2	78.5	87.6
	TiF learner w/o c	30.3	**51.0**	**74.1**	**85.0**	**91.5**	**40.3**	**57.7**	**79.1**	**90.4**	**95.9**

Table A4. N-shot accuracies on DukeMTMC-reID and VeRi-776 with seed 2. Supplementary to Table 2 (seed 1).

	Method	DukeMTMC-reID					VeRi-776				
		1	2	4	8	16	1	2	4	8	16
CLIP	CoOp [12]	6.8	10.7	16.9	27.4	33.9	10.2	12.7	18.7	31.0	32.5
	Co-CoOp [11]	7.9	12.4	19.5	32.4	42.1	29.5	30.4	32.9	36.4	37.8
	MaPLe [3]	14.7	31.6	41.2	54.3	61.6	34.8	39.6	44.5	56.4	67.9
OC	Linear-probe [7]	8.9	16.4	38.5	52.2	64.9	14.0	28.9	50.5	63.1	67.2
	Tip-Adapter [9]	29.3	37.4	47.8	60.7	66.8	36.4	48.7	58.9	72.7	79.1
	Tip-Adapter-F [9]	29.6	32.2	54.6	75.2	85.8	37.8	48.9	62.9	79.2	88.3
	TiF learner w/o c	**36.4**	**52.2**	**73.4**	**82.9**	**91.1**	**42.3**	**61.4**	**77.3**	**91.1**	**96.0**

Table A5. N-shot accuracies on DukeMTMC-reID and VeRi-776 with seed 3. Supplementary to Table 2 (seed 1).

A2.3. Other Details

Attention Map. We used the code in https://github.com/google/prompt-to-prompt/blob/main/null_text_w_ptp.ipynb, which modifies Algorithm 1 in [1] by using the unconditional embedding identified by [6]. In particular, the repository was originally designed for Stable Diffusion 1.4, and we adapted it for Stable Diffusion 2.0 and included it in Appendix.

Text Prompt. We used "a photo of a [V], a type of aircraft." on FGVCAircraft, "a high quality dermoscopic image of [V]." on ISIC-2019, "a photo of person [V] captured by surveillance camera." on DukeMTMC-reID, and "a photo of car [V] captured by surveillance camera." on VeRi-776 for CLIP adapter baselines and our TiF learner. For baselines, [V] is replaced by the class name. For our method, [V] is the rare token identifier if not using class

Method	FGVCAircraft					DukeMTMC-reID				
	1	2	4	8	16	1	2	4	8	16
SD 2.1	47.4	54.7	63.5	73.1	78.1	36.2	52.8	72.4	83.0	90.8
Encoder only	44.5	53.8	62.6	70.7	74.8	35.4	51.4	71.4	78.6	85.8
W/o mid-blocks	40.2	51.2	60.9	68.1	72.4	32.1	48.9	68.2	75.2	81.4
Ours	48.5	55.8	64.2	74.2	79.9	36.9	53.6	73.1	84.5	91.6

Table A6. Ablation on SD version and location of LoRA injections. "Encoder only" is when we only inject the linear layers in U-Net encoder. "W/o mid-blocks" is when we only inject the linear layers in the encoder and decoder (without the in-between layers).

Method	FGVCAircraft					ISIC2019					DukeMTMC-reID					VeRi-776				
	1	2	4	8	16	1	2	4	8	16	1	2	4	8	16	1	2	4	8	16
CLIP-Adapter	46.1	49.0	52.9	55.5	61.1	18.9	25.2	26.2	36.1	41.3	31.1	32.5	52.4	73.7	82.5	25.4	44.3	56.7	74.5	84.7
CoOp	24.5	31.1	35.4	41.2	50.5	12.7	14.3	16.7	21.9	33.4	9.8	12.3	18.4	31.2	34.6	12.5	15.6	20.2	31.1	34.4
Co-CoOp	32.5	33.8	38.2	46.7	52.6	15.0	19.0	20.3	25.5	32.3	10.5	13.6	22.7	34.0	50.1	28.7	33.5	34.0	41.5	62.1
MaPLe	30.2	38.4	48.8	51.6	57.3	16.0	18.2	18.9	24.7	27.8	28.9	37.6	43.2	75.0	85.7	18.8	25.9	46.0	71.2	80.7
TiF Learner	**48.5**	**55.8**	**65.0**	**72.1**	**80.4**	**24.1**	**27.6**	**33.8**	**37.2**	**44.7**	**36.9**	**53.6**	**73.1**	**83.7**	**91.6**	**41.9**	**60.7**	**78.2**	**91.2**	**96.8**

Table A7. N-shot accuracies of prompt-tuning and CLIP-adapter methods using *OpenCLIP* (ViT-H/14 trained on LAION-2B) compared with our TiF learner.

names. When using class names, we add the class names after [V]. On DukeMTMC-reID and VeRi-776, class names or class descriptions are not available for both baselines and our method. For CLIP adapters that ensemble CLIP prediction with adapter prediction, we set the CLIP prediction weight as 0, as CLIP prediction does not work when class description is not available.

Comparison Fairness. We use OpenCLIP trained on the same dataset as SD 2.0 to maximize the comparison fairness. Note that SD 2.0 uses the frozen text encoder from OpenCLIP ViT-H/14, hence we used OpenCLIP ViT-H/14 (and its adapters) as baselines for fair comparison.

Computing r_t. To compute the integral efficiently, we used end-point approximation of the integral, *i.e.*, aggregating $[1-\text{erf}(\gamma_t\delta^*+1)], [1-\text{erf}(\gamma_t\delta^*+2)], \ldots, [1-\text{erf}(\gamma_t\delta^*+500)]$ until the value becomes 0.

A3. Additional Results

Additional FSL Tasks. In Table A2 to A5, we ran additional FSL tasks by altering the seed for generating the few-shot training set. Note that all baselines and our method share the same split when using the same seed.

Additional Ablation. In Table A6, we tried using SD 2.1 as the pre-trained DM. This leads to slightly degraded performance. We conjecture that with the additional training of SD 2.1 on the same training data as SD 2.0, the DM overfits to the training data, hence becoming more difficult to adapt to novel FSL tasks. We also tried injecting other U-Net subsets, and observed that the best performance is obtained when injecting linear layers in U-Net encoder, mid-blocks as well as decoder.

Additional OpenCLIP Results. Please refer to Table A7.

Training Class-agnostic LoRA. In our method, we trained class-specific LoRA matrices separately on each class. We tried training a class-agnostic ones parameterized by θ on

Figure A2. Generated images by $d(;\theta)$ when training LoRA on few-shot images from all classes. Trained on a 4-shot task on FGVCAircraft.

FGVCAircraft, where we used class-specific prompts y_c containing the class name, *e.g.*, "a photo of a 707-320, a type of aircraft". Ideally, the trained DM should associate the specificity of each class to its respective y_c. However, as shown in Figure A2, we find that the generated image of the trained DM $d(;\theta)$ is sub-optimal, despite trying different LoRA ranks. We observed that the model seems to be finding common feature of all classes and tends to make the airplanes shorter (many fighter jets in the dataset are shorter compared to commercial aircraft). This leads to reduced performance, *e.g.*, 12.9% on 1-shot, 55.9% on 4-shot

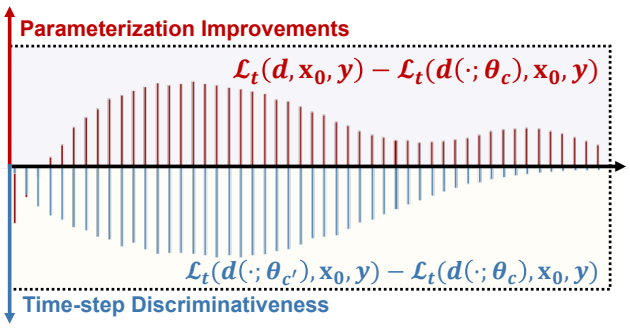

Figure A3. Top: Improvements in reconstruction loss by using parameterization (θ_c, y, t) over a pre-trained SD at each t. Bottom: The gap in reconstruction loss of using a wrong parameterization $(\theta_{c'}, y, t)$ over using the ground-truth one (θ_c, y, t).

and 76.5% on 16-shot. We leave the exploration of this approach as future work.

Parameterization Quality. We further validate our theory that connects small time-step to fine-grained attributes in Figure A3, where a small time-step corresponds to the most improvements of reconstructing fine-grained classes (after training with Eq. 6), and also corresponds to the most discriminative one to tell the ground-truth class of each test image from the rest classes.

References

[1] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. In *ICLR*, 2023. 4

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020. 3

[3] Muhammad Uzair khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *CVPR*, 2023. 4

[4] Alexander C. Li, Mihir Prabhudesai, Shivam Duggal, Ellis Brown, and Deepak Pathak. Your diffusion model is secretly a zero-shot classifier. In *ICCV*, 2023. 3, 4

[5] Ruey-Pyng Lu, Eric P Smith, and IJ Good. Multivariate measures of similarity and niche overlap. *Theoretical Population Biology*, 35(1):1–21, 1989. 2

[6] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6038–6047, 2023. 4

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 4

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, 2015. 3

[9] Renrui Zhang, Rongyao Fang, Peng Gao, Wei Zhang, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. Tip-adapter: Training-free clip-adapter for better vision-language modeling. In *ECCV*, 2022. 4

[10] Renrui Zhang, Xiangfei Hu, Bohao Li, Siyuan Huang, Hanqiu Deng, Hongsheng Li, Yu Qiao, and Peng Gao. Prompt, generate, then cache: Cascade of foundation models makes strong few-shot learners. In *CVPR*, 2023. 4

[11] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Conditional prompt learning for vision-language models. In *CVPR*, 2022. 4

[12] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. Learning to prompt for vision-language models. *IJCV*, 2022. 4