DELT: A Simple Diversity-driven `EarlyLate` Training for Dataset Distillation

Supplementary Material

Appendix

A. Limitations

Our method effectively avoids the issue of insufficient data diversity generated by *batch-to-global* methods and reduces the computational cost of the generation process. However, there is still a performance gap when training the model on our generated data compared to training on the original dataset. Also, our short-optimized data exhibits similar appearance and semantic information to the original images, which has demonstrated better privacy protection than prior train-free methods but may still potentially leak the privacy of the original dataset to some extent.

B. More Training Details

For reproducibility, we provide all our hyper-parameter settings used in our experiments in Table 1, we outline such details below.

Squeezing and Pre-trained models. Following the previous works [4, 5], we use the official PyTorch [2] pre-trained ResNet-18 model for ImageNet-1K, and we use the same official Torchvision [2] code to obtain our pre-trained models, ResNet-18 and ConvNet, for the other datasets.

Ranking. For our initialization, we simply use ResNet-18 pre-trained models to rank and select the medium images as initialization for all our datasets, except for ImageNet-100 where we simply extract the medium images based on the rankings of the original ImageNet-1K.

Recovery. For our synthetic stage, we provide the details of general hyper-parameters used for different datasets, including ImageNet-1K, ImageNet-100, ImageNette, Tiny-ImageNet, and CIFAR10, in Table 1b. Synthesizing a single image per class, i.e., IPC = 1, is special as we cannot use *rounds*, so we apply individual numbers of iterations based on both the dataset scale and the validation teacher model as outlined in Table 1c. We also utilize the BatchNorm distribution regularization term as in SRe^{2}L [5] for Eq. 7 in the main paper to improve the quality of the generated images:

$$\mathcal{R}_{\text{reg}}\left(\tilde{\boldsymbol{x}}\right)=\sum_{l}\left|\mu_{l}(\tilde{\boldsymbol{x}})-\mathbf{BN}_{l}^{\mathrm{RM}}\right|_{2}+\sum_{l}\left|\sigma_{l}^{2}(\tilde{\boldsymbol{x}})-\mathbf{BN}_{l}^{\mathrm{RV}}\right|_{2} \quad (1)$$

where l is the index of BN layer, $\mu_l(\tilde{\boldsymbol{x}})$ and $\sigma_l^2(\tilde{\boldsymbol{x}})$ are mean and variance. $\mathbf{BN}_l^{\mathrm{RM}}$ and $\mathbf{BN}_l^{\mathrm{RV}}$ are running mean and running variance in pre-trained model at l-th layer, which are globally counted.

Validation. This includes the soft-label generation [3] as used in SRe^{2}L, post-training and evaluation. We outline such details in Table 1a. We use `timm`'s version of Ran-

Figure 1. Synthetic image visualizations on Tiny-ImageNet generated by our DELT.

Figure 2. Synthetic image visualizations on ImageNette generated by our DELT.

(a) Validation settings

config	value
optimizer	AdamW
base learning rate	0.001 (all) 0.0025 (MobileNet-v2)
weight decay	0.01
batch size	100 (IPC 50) 50 (IPC 10) 10 (IPC 1)
learning rate schedule	cosine decay
training epoch	300
augmentation	RandAugment RandomResizedCrop RandomHorizontalFlip

(b) Recovery settings

config	value
α_{BN}	0.01
optimizer	Adam
base learning rate	0.25
momentum	$\beta_1, \beta_2 = 0.5, 0.9$
batch size	100
learning rate schedule	cosine decay
recovery iteration	4,000
round iteration	500 [IPC 10, 50, 100]
initialization	top medium
augmentation	RandomResizedCrop

(c) Dataset-specific settings in recovery

config	CIFAR10	Tiny-ImageNet	ImageNette	ImageNet-100	ImageNet-1K
RandAugment (`m`)	5	4	6	6	6
RandAugment (`n`)	4	3	2	2	2
RandAugment (`mstd`)	1.0	1.0	1.0	1.0	1.0
IPC1 Recovery Iterations	2K (R18)	500 (R18)	1K (R18)	-	3K (Conv4)
	3K (R101)	500 (R101)	1K (R101)	-	-
	2K (MobileNet)	500 (MobileNet)	2K (MobileNet)	-	-
	-	1K (Conv4)	4K (Conv5)	-	-

Table 1. Hyper-parameter settings.

Initialization	SRe^{2}L + w/ Init w/o `EarlyLate`	CDA + Init w/o `EarlyLate`	CDA + w/ Init + w/ `EarlyLate` (Ours)
2×2	55.3	56.9	**58.2**$_{(+1.3)}$
3×3	55.8	56.6	**58.1**$_{(+1.5)}$
4×4	55.2	56.7	**57.4**$_{(+0.7)}$
5×5	54.6	56.5	**57.3**$_{(+0.8)}$

Table 2. Performance comparison w/ and w/o `EarlyLate` on ImageNet-1K under IPC 50.

Order	DELT
Random	67.9
Ascending	67.2
Descending	67.7
Our DELT	68.2

Table 3. Impact of using different ordering on ImageNet-100 when having the same initialized images of the median probability.

Selection Strategy	DELT
Random	67.7
Ascending	66.9
Descending	67.3
Our DELT	68.2

Table 4. Comparison of the performance of different initialization strategies. The initialized images are different.

dAugment [1] with different settings depending on the synthesized dataset being validated, as shown in Table 1c.

C. More Visualization

We provide more visualizations on synthetic Tiny-ImageNet, ImageNette and CIFAR-10 datasets in Fig. 1, 2, 3. In each figure, each column represents a different class, with images progressing from long optimization at the top to short optimization at the bottom.

D. More Ablation

Performance comparison w/ and w/o `EarlyLate`. We compare using the recent CDA and SRe^{2}L as the base frameworks. The performance comparison for IPC 50 on ImageNet-1K is shown in Table 2. It can be observed that the proposed *EarlyLate* strategy enhances the performance by around 1% with the initialization.

Comparison of random, ascending and descending orders by patch probability. In our DELT, we select the N patches with scores around the median from the teacher, where the score represents the probability of the true class. To order them, we start with the median, and we go back

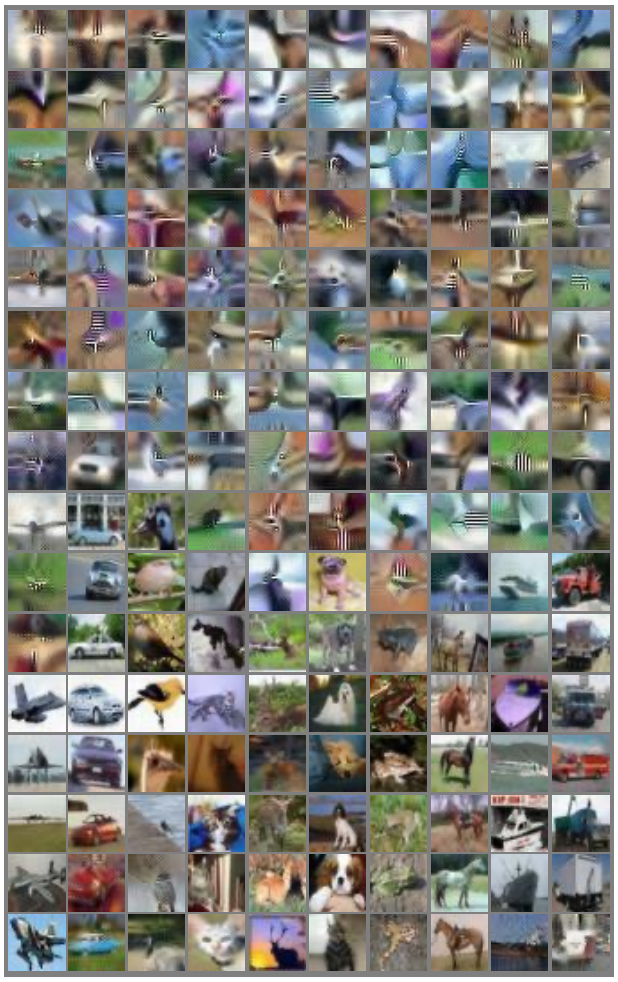

Figure 3. Synthetic images on CIFAR-10 generated by our DELT.

and forth expanding the window around the median until we cover the number of IPCs, refer to Fig. 5 of the main paper for details. The rationale is that these patches present a medium difficulty level for the teacher, allowing more potential for information enhancement through distillation gradients while having a good starting point of information. We empirically validate it by comparing different strategies in Table 4b of the main paper.

As shown in Table 3, we present the impact of using different ordering on ImageNet-100 when having the same initialized images, those around the median. We also include a comparison of different initialization strategies based on the order in Table 4. Unlike Table 3, the initialized images here are different across different strategies.

Different initial crop ranges in random crop augmentation for our DELT method. Table 5 compares different initial crop ranges in random crop augmentation for our DELT method. As shown in the results, the 0.08-1.0 range yields the best performance, which is the ablation and support for the default setting in our framework.

Random Crop Range	Top 1-acc
0.08-1.0	**67.8**
0.2-1.0	67.3
0.5-1.0	66.3
0.8-1.0	66.3

Table 5. Different initial crop ranges in random crop augmentation for our DELT method.

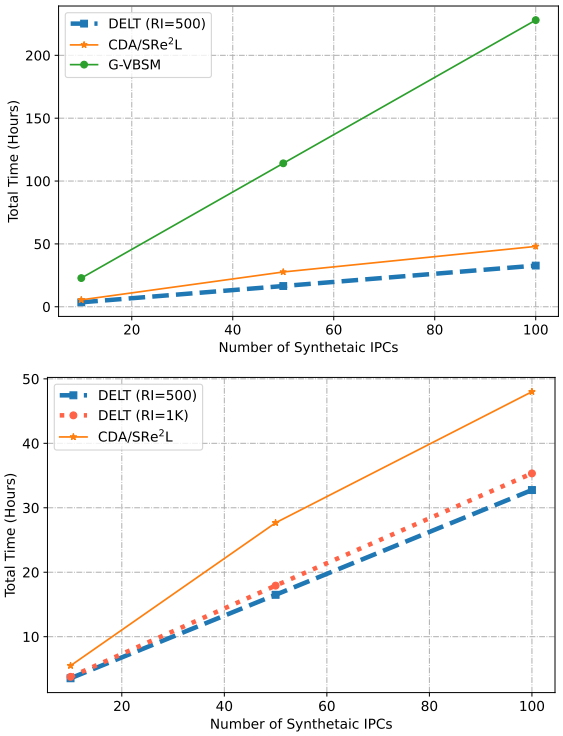

Figure 4. Visualization of computation time consumption on our DELT and other methods including CDA, SRe^{2}L, and G-VBSM.

E. Computational Efficiency

Fig. 4 illustrates the computation time required for our DELT method compared to other methods, including CDA, SRe^{2}L, and G-VBSM, at various numbers of IPCs. The top subfigure shows a comparison of DELT (with a RI of 500), CDA/SRe^{2}L, and G-VBSM, where G-VBSM demonstrates the highest computation time, scaling significantly as the number of IPCs increases. In contrast, both DELT and CDA/SRe^{2}L maintain relatively low and consistent computation times, with DELT slightly outperforming CDA/SRe^{2}L. The bottom subfigure further compares DELT with RIs of 500 and 1,000 against CDA/SRe^{2}L, highlighting that both configurations of DELT offer lower or comparable computation times to CDA/SRe^{2}L across IPC values, with minimal increase as the IPC count rises. These results emphasize DELT's efficiency in computation time, particularly in comparison to G-VBSM, making it a computationally efficient choice for scenarios with larger datasets.

References

[1] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020. 2

[2] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 1

[3] Zhiqiang Shen and Eric Xing. A fast knowledge distillation framework for visual recognition. In *European Conference on Computer Vision*, pages 673–690. Springer, 2022. 1

[4] Zeyuan Yin and Zhiqiang Shen. Dataset distillation via curriculum data synthesis in large data era. *Transactions on Machine Learning Research*. 1

[5] Zeyuan Yin, Eric Xing, and Zhiqiang Shen. Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. In *NeurIPS*, 2023. 1