Supplementary Material for Distill Gold from Massive Ores: Bi-level Data Pruning towards Efficient Dataset Distillation

We provide the following details and analyses in the supplementary:
Sec. 1: Relation between Loss and Its Derivative.
Sec. 2: Error of ITE Estimation.
Sec. 3: Comparison to Coreset Selection Methods.
Sec. 4: Overfitting Analysis.
Sec. 5: Model Generalization.
Sec. 6: Implementation Details.
Sec. 7: More Visualizations.
Sec. 8: Licenses.

1 Relation between Loss and Its Derivative

In BiLP, we use $\ell(u_r, y_r)$ as preemptive selection criterion since it is monotonous concerning $\|\frac{\partial \ell(u_r,y_r)}{\partial u_r}\|$ for common loss functions like Mean Squared Error (MSE) or cross-entropy, or say $\ell(u_r, y_r)$ and $\|\frac{\partial \ell(u_r,y_r)}{\partial u_r}\|$ are positive correlated. Let's consider the two common loss functions mentioned:

1. **Mean Squared Error (MSE)**: The MSE loss function is defined as $\ell(u, y) = \frac{1}{2}(u-y)^2$, where u is the predicted value and y is the actual value. The gradient of the MSE with respect to u is $\frac{\partial \ell}{\partial u} = u - y$. The gradient norm is then $\|\frac{\partial \ell}{\partial u}\| = |u - y|$. Thus, larger $\|\frac{\partial \ell}{\partial u}\|$ leads to larger$\ell(u, y)$.
2. **Cross-Entropy**: The cross-entropy loss function for binary classification is defined as $\ell(\boldsymbol{u}, y) = -\log(u_y)$, where $\boldsymbol{u}$ is the predicted logits of the N class and each component $u_i > 0$, and y is the actual class. The gradient of the cross-entropy with respect to each component u_i is $\frac{\partial \ell}{\partial u_i} = \begin{cases} -\frac{1}{u_y}, & \text{if } i = y \\ 0, & otherwise \end{cases}$. The gradient norm is $\|\frac{\partial \ell}{\partial u}\| = \frac{1}{u_y\sqrt{N}}$. So larger $\|\frac{\partial \ell}{\partial u}\|$ indicating a small u_y and therefore the $\ell(\boldsymbol{u}, y)$ is large.

Overall, for these two loss functions, $\|\frac{\partial \ell}{\partial u_r}\|$ and $\ell(u_r, y_r)$ are positively correlated.

2 Error of ITE Estimation

In BiLP, we have implemented the Taylor approximation technique to enhance the efficiency of ITE value computations. To quantitatively assess the error of the Taylor approximation, we conducted an experiment where both the original

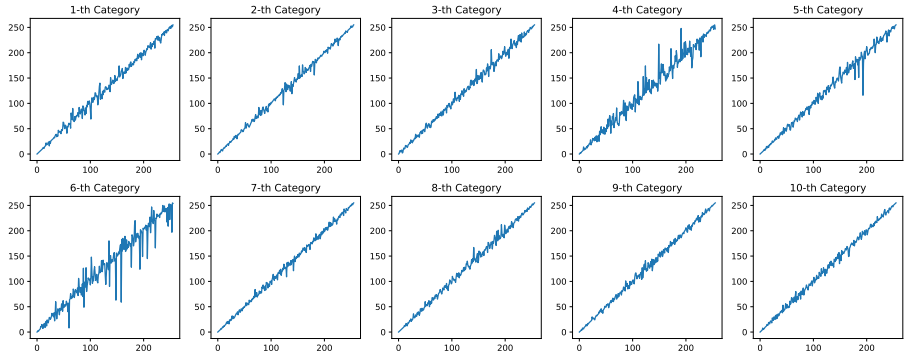


Fig. 1: Visualization of the original and the Taylor-approximated ITE rankings for CIFAR10 [11] with IPC=10. An ideal approximation would reveal a 45-degree diagonal line.

Table 1: Comparison to coreset selection on CIFAR10, IPC=1 and DC algorithm.

Selection Criterion	Pruning Ratio						
	99%	97%	95%	90%	80%	70%	0% (Full dataset)
Random	25.6±0.6	27.6±0.8	27.6±0.6	28.2±0.3	28.5±0.4	28.7±0.3	28.3±0.5
Loss (remove large)	**29.7±0.1**	**29.7±0.0**	**30.0±0.1**	**30.0±0.2**	**30.2±0.1**	**30.2±0.2**	
Coreset: CRAIG [17]	25.4±0.6	27.8±0.2	28.8±0.5	28.6±0.4	29.0±0.1	29.0±0.2	
Coreset: GradMatch [8]	26.5±0.5	28.0±0.3	28.7±0.5	28.4±0.3	28.9±0.2	29.4±0.3	
Coreset: GLISTER [9]	23.8±0.5	26.9±0.2	23.5±0.1	21.3±0.6	24.0±0.9	24.9±0.8	

and the Taylor-approximated ITE values were calculated within a mini-batch consisting of N samples. The samples were then sorted based on their ITE values, resulting in rankings represented by r_i and r'_i for each i^{th} sample within the range of $[1, N]$. To evaluate the discrepancy between the two sets of rankings, we utilized the average relative error metric, defined as $\frac{1}{N(N-1)} \sum_{i=1}^{N} |r_i - r'_i|$, with a range of $[0, 1)$. Experimenting on the CIFAR10 dataset with an IPC of 10, our analysis revealed an average error rate of 2.7%, which is considered negligible for pruning.

Furthermore, to provide a more intuitive understanding of our approximation's performance, we visualize the rankings in Fig. 1. The x-axis corresponds to the original ITE rankings, while the y-axis depicts the rankings derived from the Taylor approximation, with each of the 10 classes distinctly plotted. The closer the distribution of points to a 45-degree diagonal line, the higher the similarity between the two computation methods. The figure demonstrates that our Taylor approximation closely mirrors the original ITE rankings, thereby validating its efficacy.

Table 2: Comparison to coreset selection on CIFAR10, IPC=50 and DC algorithm.

Selection Criterion	Pruning Ratio				
	70%	60%	50%	30%	0% (Full)
Random	53.0±0.2	53.6±0.3	54.0±0.5	54.2±0.3	54.1±0.3
Loss (remove large)	**54.1±0.2**	**54.9±0.3**	**55.3±0.3**	**56.0±0.2**	
Coreset: CRAIG [17]	48.8±0.4	48.8±0.2	49.0±0.4	49.1±0.4	
Coreset: GradMatch [8]	49.0±0.3	49.1±0.3	49.1±0.4	49.1±0.5	
Coreset: GLISTER [9]	41.9±0.4	43.9±0.5	44.0±0.5	47.0±0.5	

Table 3: Maximum pruning ratio on more distillation algorithms.

Dataset	IPC	CAFE [22]	LinBa [4]	IDC [10]
CIFAR10 [11]	1	85%	30%	50%
	10	89%	70%	90%
SVHN [18]	1	70%	50%	40%
	10	40%	40%	70%
MNIST [14]	1	90%	70%	99.5%
	10	1%	60%	60%

Table 4: Maximum pruning ratio of various initializations on CIFAR10 [11].

IPC	Init	DC [27]	DM [26]
1	Noise	90%	90%
	Real	90%	85%
	Herd	90%	85%
10	Noise	70%	70%
	Real	70%	60%
	Herd	70%	70%

Table 5: Maximum pruning ratio of various networks on CIFAR10 [11].

IPC	Net	DC [27]	DM [26]
1	Conv [23]	90%	85%
	MLP	97%	95%
	ResNet [6]	95%	85%
	VGG [20]	90%	95%
	AlexNet [12]	95%	95%
10	Conv [23]	70%	60%
	MLP	60%	60%

3 Comparison to Coreset Selection Methods

The existing coreset selection methods can also be exploited as sample selection criteria. So we conduct a comparison with recent coreset selection methods on CIFAR10 [11] and DC [27] algorithm, including CRAIG [17], GradMatch [8] and GLISTER [9]. We adopt these methods as a preemptive pruning criterion. We use the algorithms implemented by the CORDS package. The results are shown in Tab. 1 and 2. CRAIG and GradMatch coreset selection can achieve a 97% maximum pruning ratio when IPC=1, though still worse than the loss criterion. The GLISTER algorithm does not perform well on dataset distillation and is worse than random selection. Thus, on the data pruning for dataset distillation, our loss indicator can surpass some sophisticated selection algorithms.

3.1 Data Redundancy on Various Architectures and Initialization

In the main paper, we examine the real data redundancy in dataset distillation by randomly removing some real samples before the training. Here we present more results in Tabs. 3 to 5 with more algorithms and various initialization and network architectures, which exhibit large pruning ratios and indicate significant data redundancy.

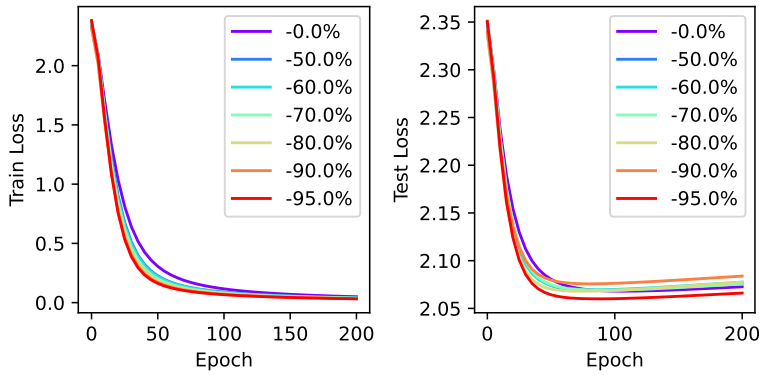


Fig. 2: Train and test loss curves for different pruning rates.

4 Overfitting Analysis

We compare the train and test loss curve on distilled data with different pruning rates in Figure 2 on CIFAR10 and IPC=1 (5 repeats). The test loss curves do not drastically increase after the convergence of training, and all the loss curves are similar, showing that a large pruning rate does not enhance overfitting problems. This is consistent with the explanation in Sec. 4.1: due to the limited capacity of the synthetic dataset, removing some unimportant or outlier data samples does not harm the distillation process.

5 Model Generalization

5.1 Cross-Architecture Generalization of Preemptive Pruning

We conducted a cross-architecture evaluation to verify whether the data pruning harms the generalization ability. We first follow DC [27] to experiment on MNIST and IPC=1. We remove the training samples with the largest loss values and we compare the training subsets with 100% (full data, original setting in DC paper), 10%, 5%, and 3%. The results are shown in Tab. 6 in the rebuttal PDF file, showing that pruning the training dataset does not damage the generalization ability of the distilled data. On the contrary, in most slots (28/36), data pruning can even enhance the generalization ability.

We also conduct experiments on larger IPCs with DC [27] and MTT [2]. The results are shown in Tab. 7 and 8. On larger IPCs, pruning the training dataset still does not damage the generalization ability of the distilled data.

5.2 Cross-Architecture Generalization of Full BiLP

In this study, we present a cross-architecture evaluation that demonstrates the efficacy of synthetic data trained on the ConvNet-D3 architecture while being assessed across different neural network architectures including ResNet [6],

Table 6: Cross-architecture generalization on CIFAR10 [11] and IPC=1 with DC [27] algorithm. Best results are marked in **bold green**. The generalization ability of pruned data is superior or comparable to the originals.

Evaluate Network	Pruning Ratio	Distill Network MLP	ConvNet	LeNet	AlexNet	VGG	ResNet
MLP	0% (Full)	24.3±1.8	26.5±0.2	24.1±0.4	26.5±0.1	25.1±0.3	24.9±0.4
	60%	23.0±1.4	26.6±0.3	24.2±0.5	26.6±0.1	25.1±0.1	**25.1±0.1**
	80%	23.7±0.9	26.5±0.4	24.3±0.7	26.6±0.1	25.2±0.1	24.6±0.3
	95%	**25.1±1.4**	**26.8±0.2**	**24.3±0.2**	**26.6±0.2**	**25.3±0.3**	24.8±0.4
ConvNet [23]	0% (Full)	22.7±1.1	28.8±0.4	21.8±0.2	27.0±0.5	25.8±0.4	26.2±0.3
	60%	23.0±1.7	28.7±0.1	22.1±0.3	26.9±0.2	25.8±0.3	26.2±0.2
	80%	23.4±0.9	28.8±0.2	22.2±0.3	26.8±0.3	25.6±0.2	**26.2±0.2**
	95%	**23.7±0.8**	**29.1±0.2**	**22.3±0.3**	**26.9±0.4**	**25.8±0.1**	26.1±0.3
LeNet [14]	0% (Full)	16.0±2.6	17.9±0.8	14.9±1.1	15.7±1.5	18.2±1.3	16.6±1.1
	60%	16.8±1.9	17.6±0.8	**15.6±0.2**	16.1±0.7	18.8±0.4	16.4±1.1
	80%	**18.9±1.5**	17.3±0.6	14.0±0.6	**16.2±1.1**	**19.3±0.6**	15.8±0.9
	95%	17.9±1.0	**17.9±1.2**	15.1±0.7	15.6±0.6	19.1±0.8	**16.8±1.2**
AlexNet [12]	0% (Full)	20.8±0.5	22.9±0.3	22.7±0.6	21.1±0.2	22.7±0.2	22.3±0.2
	60%	19.4±1.2	22.7±0.4	23.0±0.4	21.1±0.4	22.7±0.1	22.3±0.2
	80%	21.2±0.6	**23.0±0.3**	23.1±0.7	20.6±0.1	22.6±0.1	22.2±0.3
	95%	**21.3±0.4**	22.9±0.2	**23.1±0.4**	**21.2±0.5**	**22.8±0.2**	**22.3±0.2**
VGG [20]	0% (Full)	20.3±3.1	15.7±0.6	20.4±0.8	23.6±0.7	18.0±0.8	**18.5±0.4**
	60%	20.2±1.6	**15.8±0.4**	20.3±0.5	**24.0±0.6**	**18.6±0.4**	18.1±0.3
	80%	19.9±2.2	15.3±0.8	**20.6±0.8**	23.6±0.6	18.1±0.7	18.2±0.7
	95%	**20.4±1.7**	15.3±0.7	20.4±0.8	23.0±0.8	18.0±0.9	18.2±0.7
ResNet [6]	0% (Full)	18.3±1.6	16.7±0.9	15.1±0.8	17.5±0.5	16.0±0.7	18.1±1.1
	60%	**18.9±1.8**	**17.2±1.0**	**15.2±0.5**	**17.8±0.6**	**15.6±1.0**	18.5±0.6
	80%	17.0±1.1	16.3±0.7	13.9±1.0	17.6±1.3	15.2±0.6	**19.2±0.8**
	95%	18.8±1.4	16.7±0.6	14.3±0.7	17.0±1.4	15.0±0.4	18.3±0.7

EfficientNet [21], and DenseNet [7]. The experiments are conducted using the CIFAR10 dataset at three IPC levels 1, 10, and 50. As shown in Tab. 9, the synthetic data generalize well when applied to other network architectures, *e.g.* at IPC=50, all three model types achieve performance that is on par with those obtained from the in-situ evaluation using ConvNet-D3. This indicates that the synthetic data is capable of maintaining its predictive accuracy across diverse architectures. Furthermore, our comparative analysis reveals that the BiLP consistently outperforms IDC in the majority of the conducted experiments.

5.3 Generalization to Larger Data

We add experiments on larger dataset (*e.g.* ImageNet) and larger IPC (we adopt DATM [5] algorithm). We re-implement DATM with TESLA for efficiency and compare BiLP (preemptive pruning 10% data) to the reproduced DATM results. As shown in Table 10, BiLP could enhance DATM on small IPCs and is comparable at large IPCs, and could also achieve lossless performance at IPC=1000.

Table 7: Cross-architecture generalization on CIFAR10 [11] and IPC=50 with DC [27] algorithm.

Sample Ratio	Evaluate Network					
	MLP	ConvNet	LeNet	AlexNet	VGG	ResNet
0% (Full)	28.01±0.40	54.02±0.51	28.12±2.16	29.48±0.58	39.44±0.67	22.72±1.08
30%	29.48±0.28	**55.96±0.40**	30.83±1.51	29.54±2.60	41.99±0.47	24.35±0.42
50%	**30.40±0.28**	55.25±0.32	31.15±1.25	30.53±2.21	43.09±0.41	**25.81±1.04**
70%	30.15±0.21	54.77±0.47	**31.44±0.72**	**33.45±1.18**	**43.35±0.60**	25.48±0.53

Table 8: Cross-architecture generalization on CIFAR10 [11] and IPC=10 with MTT [2] algorithm.

Sample Ratio	Evaluate Network			
	ConvNet	AlexNet	VGG	ResNet
0% (Full)	64.3±0.7	34.2±2.6	50.3±0.8	46.4±0.6
10%	**64.6±0.4**	**34.3±2.4**	**51.1±1.1**	**48.6±0.4**

We also conduct experiment of SRe2L [24] in Table 11. BiLP could enhance the performance with less data, *e.g.*, BiLP with only 50% data could significantly enhance SRe2L.

6 Implementation Details

6.1 Datasets and Metric

Our experiments are conducted on the following datasets and we report the top-1 accuracy as the metric, most of which are widely adopted in dataset distillation.

- CIFAR10 [11]: image dataset of common objects with 10 classes and 50,000 image samples. The images are 32x32 with 3 channels.
- CIFAR100 [11]: image dataset of common objects with 100 classes and 50,000 samples. The images are 32x32 with 3 channels.
- SVHN [18]: street digit dataset with 10 classes and 73,257 samples. The images are 32x32 with 3 channels.
- TinyImageNet [13]: a subset of ImageNet with 200 classes and 100,000 images. The images are 64x64 with 3 channels.
- ImageNet [3]: image datasets of common objects with 1000 classes and 1,281,167 samples. We resize the images to 64x64 with 3 channels following the previous setting [29].
- Kinetics-400 [1]: human action video dataset with 400 classes and 215,617 video samples. The videos are resampled to 8 frames per clip and resized to 64x64.

Table 9: Cross-architecture evaluation of synthetic data trained on ConvNet-D3 with BiLP, on CIFAR10 [11].

Architecture	Method	IPC 1	IPC 10	IPC 50
ConvNet-D3 [23]	IDC	50.6±0.4	67.5±0.5	74.5±0.1
	BiLP+IDC	**51.5±0.3**	**69.4±0.5**	**75.4±0.2**
ResNet [6]	IDC	42.8±1.2	64.8±0.9	71.9±1.3
	BiLP+IDC	**43.3±0.6**	**65.5±1.0**	**72.8±0.4**
EfficientNet [21]	IDC	38.5±1.0	42.2±0.8	71.4±1.5
	BiLP+IDC	**38.5±0.6**	**43.7±2.5**	**71.9±1.0**
DenseNet [7]	IDC	37.9±0.6	**64.8±0.8**	70.4±0.5
	BiLP+IDC	**38.3±0.5**	64.6±0.4	**70.7±0.7**

Table 10: DATM [5] and BiLP performance on CIFAR10. Full=84.8%.

IPC	1	10	50	500	1000
DATM	47.0	65.7	72.9	**81.3**	**84.8**
DATM+BiLP	**47.4**	**66.1**	**74.0**	**81.3**	84.6

Table 11: SRe2L [24] and BiLP performance on large dataset ImageNet [3].

IPC	1	10
SRe2L	1.2	21.3
SRe2L+BiLP (50% data)	**1.6**	**27.0**

6.2 Network Architectures

Following the previous work, in most of the experiments, we adopt ConvNetD3 as the network to probe the data. This network consists of 3 convolutional layers with a 3x3 filter, each of which has 128 channels and is followed by a ReLU non-linearity and an InstaceNorm layer. The average pooling layer aggregates the feature map to a 128d vector and produces the logit with a linear layer.

We also adopt other architectures, including MLP (three linear layers with hidden layer size 128), AlexNet [12], ResNet18 [6] (ResNet18+BatchNorm with average pooling for DC algorithm), and VGG11 [20] (we use VGG11+BatchNorm for DC algorithm).

6.3 Experiments of Random or Loss Selection

In Tab. 1-3 in the main paper, we extensively study the critical sample ratio by random or loss value. We mainly follow the default experiment settings given by each algorithm. The experiments are conducted on RTX 4090 GPU. We list the experiment details:

1. For DC [27] and DSA [25], on all datasets, we run the distillation for 1000 iterations with SGD optimizer and momentum 0.5. The number of inner loop and outer loop are (1, 1) for IPC=1, (10, 50) for IPC=10, (50, 10) for IPC=50. The learning rate of synthetic image and network are 0.1 and 0.01. The batch size for each class is 256 and when the sample ratio is low, we half the batch size until it is less than twice the largest class size.

We use `color, crop, cutout, scale, rotate` DSA augmentation on all datasets and additional `flip` on the non-digit datasets. By default, `noise` initialization is used.

2. For DM [26], we run the distillation for 10000 iterations on TinyImageNet and 20000 iterations for the others with SGD optimizer and momentum 0.5. The learning rate of synthetic image and network are 1.0 and 0.01. The batch size for each class is 256 and when the sample ratio is low, we half the batch size until it is less than twice the largest class size. The same Siamese augmentation strategy is used as in the DSA experiments. By default, `real` initialization is used (the initial images are drawn after dropping).
3. For MTT [2], we drop the same data samples for buffering and distillation. The expert trajectories are trained for 50 epochs for 100 repeats and we run the distillation for 10000 iterations. We appreciate and follow the detailed hyper-parameters provided by the authors.
4. For CAFE [22], as default, we run the distillation for 2000 iterations. The initial learning rate is 0.1 and decays by 0.5 at 1,200, 1,600, and 1,800 iterations. The weight of the inner layer matching loss is 0.01 and an additional loss weight of 0.1 is put on the matching loss of the third and fourth layers. `Noise` initialization is used.
5. For LinBa [4], we run distillation for 5000 iterations with SGD optimizer with momentum 0.9. The inner steps of BPTT are 150 and the number of bases is 16. The learning rate of synthetic image and network is 0.1 and 0.01.
6. For IDC [10], we use the "reproduce" setting of the opened source code, which automatically sets up the tuned hyper-parameters. We use multi-formation factor 2.
7. For KIP [19], we test on the finite-width model (KIP-NN) and use label learning. We use longer training steps for converged results.
8. For FRePo [29], we use the official PyTorch implementation and the default parameters, except the learning rate of 0.001 and we run the distillation for 500,000 steps.
9. For HaBa [15], we follow the official instructions and adopt the parameters from MTT. And for the exclusive parameters for HaBa, we use the values given in the code.
10. For RFAD [16], we test on the finite-width model (ConvNet) and load the training hyperparameters for finite training results in the paper. The choice of label learning follows the remarks in the paper.
11. For IDM [28], we thank the authors and we directly adopt the official running commands.

The removal of data samples is class-wise. Each experiment is repeated 5 times for mean μ and standard deviation σ and we regard the experiment performance as comparable to the experiment on full data if its mean accuracy is within the $[\mu - \sigma, \mu + \sigma]$ of full data performance.

Table 12: Hyper-parameters of BiLP in different experiments.

Dataset	CIFAR10 [11]			CIFAR100 [11]		SVHN [18]		
IPC	1	10	50	1	10	1	10	50
Preemptive pruning rate α	0.1	0.3	0.1	0.6	0.4	0.2	0.3	0.3
Adaptive pruning rate β	0.3	0.3	0.3	0.3	0.3	0.2	0.2	0.2
Data update frequency (lazy selection)	10	5	5	10	10	10	5	5

6.4 Computation of Empirical Loss Criterion

The parameters and settings of the empirical loss criterion for the preemptive pruning are as follows: We train the ConvNetD3 model on each dataset (ConvNetD3+GRU for Kinetics-400) for multiple trials for the loss indicator. We take the average loss curve of multiple trials. By default, we use a Gaussian filter with $\sigma = 3$ to smooth the loss curve and take the loss value at the last epoch, which is approximately equivalent to the **weighted mean loss value of the last 8 epochs**. The training details are:

- CIFAR10: 50 trials for 100 epochs with learning rate 3.0e-3 and batch size 512.
- CIFAR100: 50 trials for 250 epochs with learning rate 5.0e-3 and batch size 512.
- MNIST: 50 trials for 50 epochs with learning rate 3.0e-4 and batch size 512.
- SVHN: 50 trials for 100 epochs with learning rate 1.0e-3 and batch size 512.
- TinyImageNet: 50 trials for 100 epochs with learning rate 5.0e-3 and batch size 512.
- ImageNet: 30 trials for 20 epochs with learning rate 3.0e-3 and batch size 256 (early stop).
- Kinetics-400: 10 trials for 20 epochs with learning rate 1.0e-2 and batch size 128 (early stop).

Note that considering the conclusion in Sec. 5.5, we have adopted an early stop on large-scale datasets to reduce the training cost.

6.5 Experiments of BiLP

We use 3-layer ConvNet for CIFAR and SVHN datasets, and 4-layer ConvNet for TinyImageNet. For BiLP, the momentum for running stats of ITE is set to 0.1 and we set $\beta = 30\%$ by default. The other hyperparameters vary among different datasets and please refer to the supplementary. We take the mean and standard deviation of the accuracy of 5 random trials. Specifically, we show the hyper-parameters of different experiment settings in Tab. 12. We use the same parameters for multi-formation factor 2 or 3 for IDC.

6.6 Experiments on the Large-scale Datasets

We apply our selection paradigm on larger-scale datasets in Sec. 5.2 in the main paper. The experiments are conducted on at most 4 RTX 3090 GPUs and the details are as follows:

- ImageNet, DC: the training of DC exceeds the usual GPU capacity so in compromise we separate the 1000 classes into ten 100 class splits, which will slightly decrease the accuracy. The other hyper-parameters are the same as the previous experiments. For our paradigm, we prune 50% samples and early stop at 800 iterations due to its faster convergence.
- ImageNet, DM: the DM algorithm is safe for class-separate training so we separate the classes into 4 splits at IPC=1, 8 splits at IPC=10, and 20 splits at IPC=50. We run the distillation for 5000 iterations with a learning rate of 5.0. For our paradigm, we prune 50% samples and early stop at 2,000, 3,000, and 3,000 iterations for IPC=1/10/50 respectively.
- ImageNet, MTT: the expert trajectory is too expensive to compute so we only run MTT with our selection method. We prune 90% samples which reduces 84% of the trajectory training time. We train 60 trajectories for 50 epochs. MTT also requires large GPU memory due to the unrolling of back-propagation, so we use synthetic steps=5, expert epochs=2, and maximum start epoch=5. We run the distillation for 5000 iterations with an image learning rate of 30,000 and a step size learning rate of 1.0e-6.
- Kinetics-400, DM: on Kinetics, we run DM for 5000 iterations with a learning rate of 5.0 and batch size of 128. We separate the classes into 8 splits at IPC=1 and 20 splits at IPC=10. For our paradigm, we prune 50% samples and early stop at 4000 iterations. We do not use DSA augmentation for Kinetics.
- Kinetics-400, MTT: we prune 90% samples and train 40 trajectories for 50 epochs with batch size 128. We use synthetic steps=5, expert epochs=2, and maximum start epoch=5. We run the distillation for 5000 iterations with an image learning rate of 30,000, step size learning rate of 1.0e-6, real batch size 128, and synthetic batch size 64. We do not use DSA augmentation for Kinetics.

7 More Visualizations

In this section, we present some data samples at different loss levels to qualitatively visualize the selection criterion.

We first stratify various datasets into 10 layers according to per-sample loss values and visualize some samples in the layer with the smallest or largest utility in Fig. 3, including the large-scale datasets (ImageNet and Kinetics-400). As shown in the figure, the samples with small loss are noisy and usually hard and corner cases, *e.g.* only part of the birds are shown, some dogs are acting in strange poses, or the images of ships are captured with unusual viewing angles.

Meanwhile, the samples with large losses are easy cases that have ideal saliency, viewing angle, and clean background.

The digit datasets (MNIST, SVHN) show significantly more *diversity vanishing* than the rest of the realistic datasets. Moreover, the diversity vanishing issue is mild for large-scale datasets such as ImageNet since the intra-class discrepancy is large such that any subset is diversified enough.

To extend our discussion on the data diversity (Sec. 5.4 in the main paper), we give some more examples to compare the diversity for data strata with different loss values in Fig. 4 on MNIST. The groups with large loss values are mainly corner cases. Furthermore, as the loss value decreases (S7 or S10), the diversity significantly drops as shown in Fig. 4 (c, d, g, h).

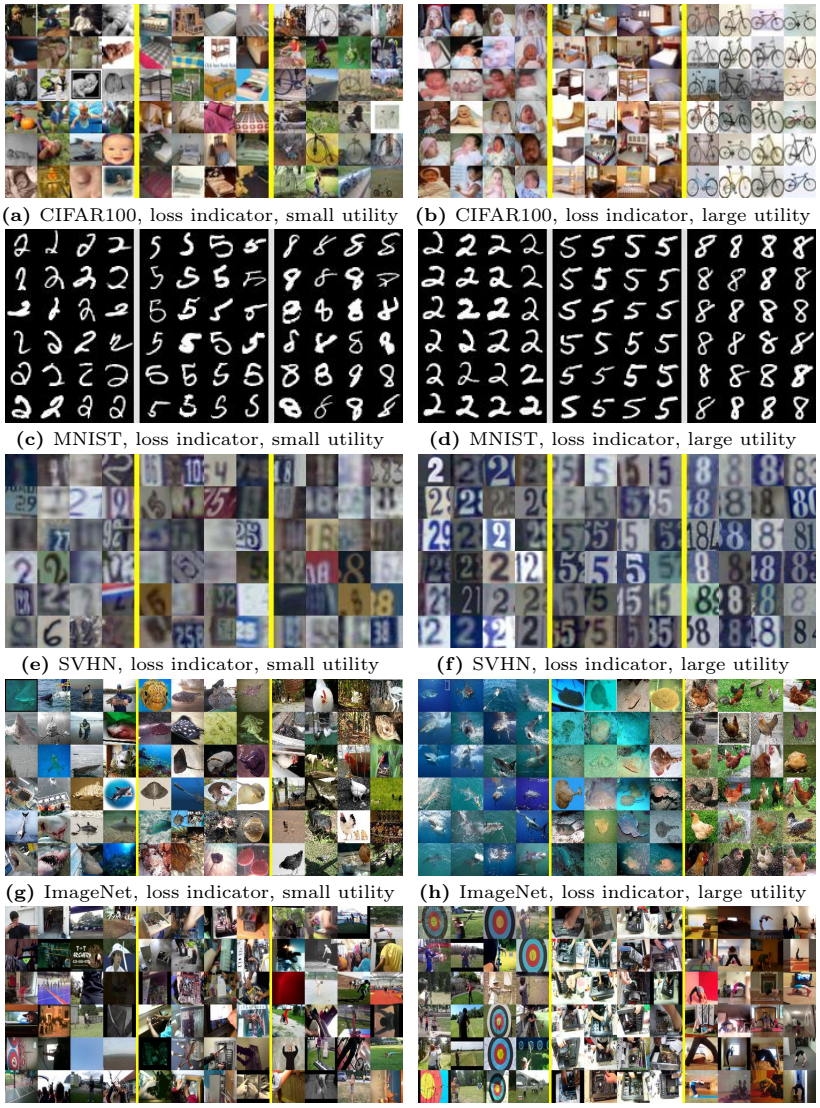
8 Licenses

Here are the source and license of the assets involved in our work. We sincerely appreciate and thank the authors and creators.
Datasets:

- CIFAR10, CIFAR100 [11]: URL, unknown license.
- MNIST [14]: URL, MIT License.
- SVHN [18]: URL, unknown license.
- Tiny-ImageNet [13]: URL, unknown license.
- ImageNet [3]: URL, custom license, research, non-commercial.
- Kinetics-400 [1]: URL, Creative Commons Attribution 4.0 International License.

Code:

- DC [27], DSA [25], DM [26]: URL, MIT License.
- MTT [2]: URL, MIT License.
- CAFE [22]: URL, no license.
- LinBa [4]: URL, no license.
- IDC [10]: URL, MIT License.
- KIP [19]: URL, no license.
- FRePo [29]: URL, no license.
- HaBa [15]: URL, Apache-2.0 License.
- IDM [28]: URL, no license.
- RFAD [16]: URL, no license.
- CORDS: URL, MIT license.



(a) CIFAR100, loss indicator, small utility **(b)** CIFAR100, loss indicator, large utility

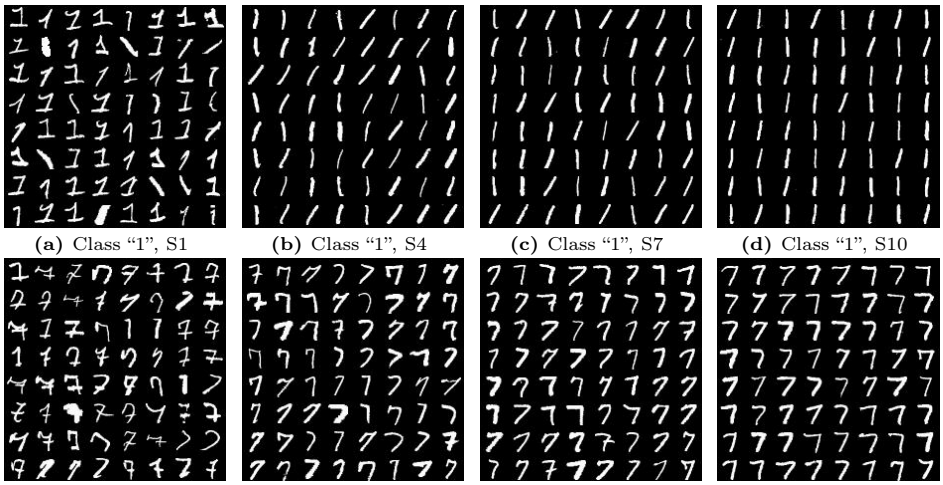
(c) MNIST, loss indicator, small utility **(d)** MNIST, loss indicator, large utility

(e) SVHN, loss indicator, small utility **(f)** SVHN, loss indicator, large utility

(g) ImageNet, loss indicator, small utility **(h)** ImageNet, loss indicator, large utility

(i) Kinetics-400, loss indicator, small utility **(j)** Kinetics-400, loss indicator, large utility

Fig. 3: Qualitative comparison of multiple datasets. We conduct stratified experiments with loss indicators and show samples in the layers with the smallest utility (left column) or largest utility (right column). We show three classes for each dataset.



(a) Class "1", S1 **(b)** Class "1", S4 **(c)** Class "1", S7 **(d)** Class "1", S10

(e) Class "7", S1 **(f)** Class "7", S4 **(g)** Class "7", S7 **(h)** Class "7", S10

Fig. 4: More examples of different strata in the MNIST dataset. The data are stratified by classification loss. The samples in S1 have the lowest loss values and those in S10 have the largest loss. The diversity significantly drops when the sample loss decreases (*e.g.* S7, S10).

References

1. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: CVPR (2017)
2. Cazenavette, G., Wang, T., Torralba, A., Efros, A.A., Zhu, J.Y.: Dataset distillation by matching training trajectories. In: CVPR (2022)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
4. Deng, Z., Russakovsky, O.: Remember the past: Distilling datasets into addressable memories for neural networks. arXiv preprint arXiv:2206.02916 (2022)
5. Guo, Z., Wang, K., Cazenavette, G., Li, H., Zhang, K., You, Y.: Towards lossless dataset distillation via difficulty-aligned trajectory matching. arXiv preprint arXiv:2310.05773 (2023)
6. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
7. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708 (2017)
8. Killamsetty, K., Durga, S., Ramakrishnan, G., De, A., Iyer, R.: Grad-match: Gradient matching based data subset selection for efficient deep model training. In: International Conference on Machine Learning. pp. 5464–5474. PMLR (2021)
9. Killamsetty, K., Sivasubramanian, D., Ramakrishnan, G., Iyer, R.: Glister: Generalization based data subset selection for efficient and robust learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 8110–8118 (2021)
10. Kim, J.H., Kim, J., Oh, S.J., Yun, S., Song, H., Jeong, J., Ha, J.W., Song, H.O.: Dataset condensation via efficient synthetic-data parameterization. In: ICML (2022)
11. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
12. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Communications of the ACM **60**(6), 84–90 (2017)
13. Le, Y., Yang, X.: Tiny imagenet visual recognition challenge. CS 231N **7**(7), 3 (2015)
14. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
15. Liu, S., Wang, K., Yang, X., Ye, J., Wang, X.: Dataset distillation via factorization. arXiv preprint arXiv:2210.16774 (2022)
16. Loo, N., Hasani, R., Amini, A., Rus, D.: Efficient dataset distillation using random feature approximation. arXiv preprint arXiv:2210.12067 (2022)
17. Mirzasoleiman, B., Bilmes, J., Leskovec, J.: Coresets for data-efficient training of machine learning models. In: International Conference on Machine Learning. pp. 6950–6960. PMLR (2020)
18. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning (2011)
19. Nguyen, T., Chen, Z., Lee, J.: Dataset meta-learning from kernel ridge-regression. arXiv preprint arXiv:2011.00050 (2020)
20. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

21. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
22. Wang, K., Zhao, B., Peng, X., Zhu, Z., Yang, S., Wang, S., Huang, G., Bilen, H., Wang, X., You, Y.: Cafe: Learning to condense dataset by aligning features. In: CVPR (2022)
23. Wang, T., Zhu, J.Y., Torralba, A., Efros, A.A.: Dataset distillation. arXiv preprint arXiv:1811.10959 (2018)
24. Yin, Z., Xing, E., Shen, Z.: Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. arXiv preprint arXiv:2306.13092 (2023)
25. Zhao, B., Bilen, H.: Dataset condensation with differentiable siamese augmentation. In: ICML (2021)
26. Zhao, B., Bilen, H.: Dataset condensation with distribution matching. In: WACV (2023)
27. Zhao, B., Mopuri, K.R., Bilen, H.: Dataset condensation with gradient matching. arXiv preprint arXiv:2006.05929 (2020)
28. Zhao, G., Li, G., Qin, Y., Yu, Y.: Improved distribution matching for dataset condensation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7856–7865 (2023)
29. Zhou, Y., Nezhadarya, E., Ba, J.: Dataset distillation using neural feature regression. arXiv preprint arXiv:2206.00719 (2022)