Dataset Distillation by Automatic Training Trajectories

Dai Liu[1], Jindong Gu[2], Hu Cao[1], Carsten Trinitis[1], and Martin Schulz[1]

[1] Technical University of Munich, Germany
[2] University of Oxford, United Kingdom

Abstract. Dataset Distillation is used to create a concise, yet informative, synthetic dataset that can replace the original dataset for training purposes. Some leading methods in this domain prioritize long-range matching, involving the unrolling of training trajectories with a fixed number of steps (N_S) on the synthetic dataset to align with various expert training trajectories. However, traditional long-range matching methods possess an overfitting-like problem, the fixed step size N_S forces synthetic dataset to distortedly conform seen expert training trajectories, resulting in a loss of generality—especially to those from unencountered architecture. We refer to this as the Accumulated Mismatching Problem (AMP), and propose a new approach, Automatic Training Trajectories (ATT), which dynamically and adaptively adjusts trajectory length N_S to address the AMP. Our method outperforms existing methods particularly in tests involving cross-architectures. Moreover, owing to its adaptive nature, it exhibits enhanced stability in the face of parameter variations. Our source code is publicly available at https://github.com/NiaLiu/ATT

Keywords: Dataset Distillation · Task-Specific Dataset Compression · Dataset Condensation

1 Introduction

Deep learning has showcased remarkable achievements across various computer vision problems [24, 43, 56]. Nevertheless, its success often hinges on extensive datasets, resulting in substantial computational demands. Recognizing the escalating computational costs [41, 47], numerous academic publications have delved into addressing these challenges. A key focus has been on mitigating the intensive training process for end-users by developing significantly reduced datasets. A conventional method for dataset reduction is Coreset Selection (CS) [4, 18, 48, 53, 62], which involves curating a subset of the most informative training samples. However, many CS methods are characterized as greedy algorithms, leading to a trade-off between speed and accuracy [2, 9, 48, 52, 64]. Additionally, the information provided by these methods is constrained by the selected samples from the original dataset. In 2018, Wang et al. [59] introduced Dataset Distillation

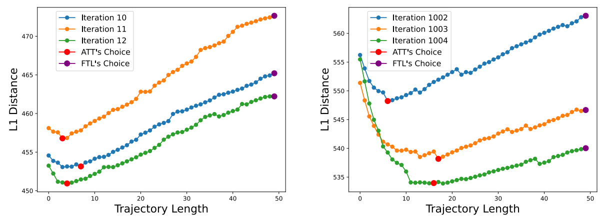

Fig. 1: The plots shows L1 distance between each network from a training trajectory and the corresponding target, and chosen target by different method. The experiments are carried on CIFAR-10. Existing methods employs Fixed Training Length (FTL), which select network at the end of a trajectory. But our method ATT dynamically selects network possessing closest distance to targets. Left plot shows examples from beginning iterations, and right plot shows higher iterations. ATT dynamically adjusts matching target, thus avoid large matching error and unwanted stretching of trajectories.

(DD), a technique that prioritizes the extraction of an informative and compact dataset capable of replacing the original training dataset in various training tasks. The technique not only provides a significantly smaller synthetic dataset for training but also conceals details in the original dataset, thereby enhancing privacy [15]. DD methods have been widely applied in various fields, including federated learning [45, 50, 57, 72, 73], continual learning [46, 59, 60], studies on attacks [36,37,59], and many other areas [7,14,26,61,71].

The DD methodologies can be classified into two distinct categories based on their alignment targets: short-range matching methods [40,58,66,68,69] and long-range matching methods [6,16,31]. Short-range matching strategies, notably initiated by a gradient-matching approach [69], concentrate on aligning a single training step executed on distilled data with that performed on the original data. Conversely, long-range matching Dataset Distillation (LDD) entail the alignment of multiple training steps. Empirical validation by Cazenavette et al. [6] demonstrates that long-range matching consumes more computation but exhibits superior performance in terms of test outcomes comparably, and attributed the great test performance to the ability to circumvent short-sightedness. Following research improved LDD further, such as with flat trajectories [16], with prunning to filter out the hidden noise [31], or investigating training stages [20].

Nonetheless, our empirical findings expose a common trend in conventional LDD methods, where inaccurate predictions are consistently reinforced throughout the majority of iterations. We attribute this issue to their reliance on Fixed Trajectory Length (FTL), which lacks the adaptability to align various steps of expert trajectories, as shown in Fig. 1. This reinforcement, driven by the FTL, results in the accumulation of matching errors over the distillation process. Importantly, these errors not only persist, but accumulate in the synthetic dataset. Consequently, the synthetic dataset tends to overly conform to observed matching targets while struggling to generalize effectively to unseen matching targets. We term this issue the AMP (Accumulated Mismatching Problem).

Our studies underscore the significance of this observation, and we propose ATT (Automatic Training Trajectories). ATT is specifically designed to dynamically and adaptively adjust matching objects and trajectory length, aiming to diminish AMP throughout the distillation process. Our method paves the way towards adaptive and dynamical matching, fostering improved generalization and accuracy in synthetic dataset distillation.

In summary, our main contributions are as follows:

- We revisit the domain of long-range matching dataset distillation, and identify the Accumulated Mismatching Problem (AMP).
- We illustrate how AMP contributes to the iterative matching error and establish a clear correlation between AMP and Accumulated Trajectory Error (ATE).
- We present an innovative Long-range Matching Dataset Distillation (LDD) method named Automatic Training Trajectories (ATT). ATT incorporates a dynamic and adaptive approach to the selection of matching objects by adjusting the matching length, leveraging a minimum distance strategy to effectively address the challenges posed by AMP.

2 Related Work

2.1 Dataset Distillation

Dataset Distillation (DD) was initially introduced by Wang et al. [59], drawing inspiration from Knowledge Distillation [23]. Existing DD baselines can be categorized into short-range and long-range matching methods based on the number of steps unrolled on the original dataset. In this context, kernel ridge regression-based methods are included under short-range matching.

Short-Range Matching Dataset Distillation (SDD): Dataset Condensation (DC), introduced by Zhao et al. [69], has been a subject of extensive research and it involves matching one-step updated gradients. Subsequent research by Zhao et al. [68] enhanced performance and reduced synthesis costs by incorporating Batch Normalization and aligning feature distributions. [70], [68] further addressed the problem of miss-alignment among classes. [30] focuses on capturing differences between classes, and [58] overcomes the bias of gradient matching methods by aligning layer-wise features. [66] enhanced DC by boosting the distillation process with Differentiable Siamese Augmentation (DSA). [63] presents a new DC framework decouples bi-level optimization, and makes a step towards real world applications. Approaches applying a kernel-based meta-learning framework with an infinitely wide neural network [40] have also sparked significant research. This includes methods implementing model pool and label learning [74], and adopting convexified implicit gradients [38].

Long-Range Matching Dataset Distillation (LDD): DD, as introduced by [59], initially focused on one-step matching. The concept of Matching Training Trajectories (MTT) was later introduced by [6] to replicate the long-range training dynamics of the original dataset. This approach overcomes the short-sightedness in single-step distillation and introduced the first long-range matching method, referred to as the vanilla LDD method in this work. Li et al. [31] implements Parameter Pruning (PP) to enforce matching on significant parameters. Additionally, [16] discovers the Accumulated Trajectory Error (ATE) in the distillation and evaluation phases, adopting flat expert trajectories to alleviate ATE, and [20] reveals different training stages contains different information.

Divergent Research Focus: Other concurrent works have improved DD baselines with various approaches. Some have adopted soft labels [5, 11, 51], while others have implemented data parameterization [14, 25, 34] and data factorization [8, 29, 32, 67]. Another approach involves encoding labels and synthetic datasets into a learnable addressing matrix [14]. Some works focus on selecting representative real samples [35], using model augmentation [65], and reducing the budget for storage and transmissions with slimmable DD [34]. Additionally, there are efforts focusing on using only a few networks [33].

2.2 Sample Selection

Like DD, instance selection [42], coreset selection [4, 18, 48], and dataset pruning [62] also aim to identify representatives from a given dataset. These sample selection methods have been widely applied in diverse fields [3, 12, 21, 39, 54]. Established techniques, such as random selection, herding methods [9, 10], and forgetting [44, 52], can be adapted as comparisons to dataset distillation methods. While similar to other selection methods, DD, in its aim of creating information-rich images that may not necessarily be considered actual samples. Multiple studies [6, 66, 69] have empirically proven that dataset distillation, especially at higher compression ratio, significantly outperforms sample selection.

3 Method

Conventional LDD methods typically unroll a fixed length of trajectory on the synthetic dataset to match expert trajectories. However, a fixed trajectory length cannot effectively handle variations within expert trajectories, leading to mismatched predictions and accumulated errors in the synthetic dataset, which we refer to the Accumulated Mismatching Problem (AMP). In this section, we first introduce the conventional LDD and the typical matching strategy involving Fixed Trajectory Length (FTL) in Sec. 3.1. Subsequently, we empirically demonstrate the AMP in conventional LDD and its impact on Accumulated Trajectory Error (ATE) in Sec. 3.2. Lastly, we present our proposed method, Automatic Training Trajectories (ATT), designed to address AMP, and provide the pseudo-code in Sec. 3.3.

3.1 LDD

Problem Description: LDD methods reveal the information within the real dataset D_T through the creation of training trajectories. Specifically, we define a training trajectory as a function $T_{D,f}(\theta_0, N)$. The function utilizes classifier f initialized with network parameters θ_0 and optimizes the classifier iteratively for a total of n steps over a dataset D. $T_{D,f}(\theta_0, N)$ outputs the difference between the network parameters at the N-th iteration and those at the initial iteration. Namely,

$$T_{D,f}(\theta_0, N) = \theta_N - \theta_0. \tag{1}$$

LDD endeavors to enhance the synthetic dataset D_S by aligning it with the expanded information in the form of trajectories. Specifically, the objective is to address the following optimization problem, as shown in the following:

$$\varDelta T = \|T_{D_S,f}(\theta_i, N_S) - T_{D_T,f}(\theta_i, N_T)\|_2^2, \tag{2}$$

$$D_S = \underset{D_S}{\arg\min} \underset{\theta_0 \sim P_{\theta_0}}{\mathbb{E}} [\varDelta T]. \tag{3}$$

We use subscripts $_S$ and $_T$ to denote parameters associated with synthetic data and real data, respectively. P_{θ_0} is the probability distribution of network initialization.

In practical scenarios, the training trajectory is unrolled on the real dataset D_T, generating expert trajectories sets as $\{\theta^*_{0,0}, \theta^*_{0,1}, \cdots, \theta^*_{0,M}, \theta^*_{1,0}, \cdots, \theta^*_{N,M}\}$. N is the number of trajectories, M is trajectory length. The expert set is precollected and stored in buffers, with additional disk storage to alleviate memory constraints.

Following the collection of the buffer, we initialize the student trajectory using any expert from the set of expert trajectories, such as $\theta'_{i,0} = \theta^*_{i,0}$. Subsequently, a similar training trajectory is unfolded on the synthetic dataset D_S for t steps, represented as $\theta'_{i,t}$. In conventional LDD methods, a fixed value N_S is chosen for t. The objective is to minimize the discrepancy between the student prediction $\theta'_{i,t}$ and the expert trajectory unrolled at N_T steps, denoted as $\theta^*_{i,NT}$. The optimization of D_S is then carried out by minimizing the defined loss function

$$L = \frac{\|\theta'_{i,N_S} - \theta^*_{i,N_T}\|_2^2}{\|\theta^*_{i,0} - \theta^*_{i,N_T}\|_2^2}. \tag{4}$$

This loss function quantifies the distance between the final state of the student trajectory and the expert trajectory using the squared L2 norm. Subsequently, the distance is normalized by the squared Euclidean distance between the initial and final states of the expert trajectory, allowing for the amplification of weak signals in the expert sets.

Fixed Trajectory Length LDD methods conventionally adopt a Fixed Trajectory Length (FTL) strategy, wherein a predetermined step size denoted as N_S is fixed for all instances of trajectory matching. This fixed step size is uniformly applied to match various experts from the expert trajectory set. Previous studies, as observed in studies such as [6,16], often conducted experiments involving an exhaustive search to determine an optimal N_S. However, our investigation

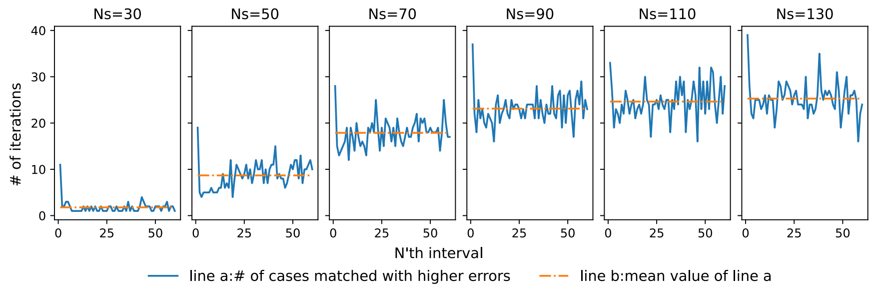

Fig. 2: The figure illustrates that Fixed Trajectory Length (FTL) matches all experts with avoidable matching error throughout the distillation process. The figure is generated from experiments conducted on CIFAR-10 with Images Per Class (IPC) set to 1. We collect the number of cases matches with larger matching errors $\|N_S - N_{opt}\| \geq \gamma$, at every 50 iterations throughout the distillation process. The number of cases matched with larger errors fluctuates over entire process. The same can be observed with the mean value of line a. From left to right, we observe the persistence of this issue across different step size N_S for FTL. Notation: #: number, N'th interval: the N'th 50 iters.

reveals that a fixed trajectory length introduces the Accumulated Mismatching Problem (AMP). This problem leads to persistent matching errors that do not diminish, even with an optimal N_S. The issue persists across variations in N_S and remains throughout the entire distillation process. Further details are discussed in the subsequent Sec. 3.2.

3.2 Accumulated Mismatching Problem (AMP)

In traditional LDD methods, a specific trajectory length denoted as N_S is fixed while matching various segments of expert trajectories. Network parameters are predicted for all instances of matching based on this FTL, namely, $\theta'_{i,N_S} = \theta'_{i,0} + T_{D_S,f}(\theta'_{i,0}, N_S)$. However, due to variations in expert trajectories arising from different steps on trajectories and different initialization of them, the designated prediction θ'_{i,N_S} lacks the necessary adjustments to accommodate such variations among different experts. This discrepancy leads to the so-called Accumulated Mismatching Problem (AMP).

We provide details in experiments conducted on CIFAR-10 with a configuration of 1 Image Per Class (IPC). The chosen baseline is the vanilla LDD method [6], to mitigate any unintended buffered effects other than FTL. In our experimental analysis, we initiate the process by generating an exhaustive set of potential predictions $pred = \{\theta'_{i,1}, \theta'_{i,2}, \cdots \theta'_{i,N_S}\}$ under diverse values of N_S ranging from 30 to 130.

Subsequently, we measure the mean square error between each prediction and the corresponding target. The $pred$ associated with the minimal error is identified at $\theta'_{i,N_{opt}}$, at the optimal step N_{opt}. The existance of the gap between N_{opt} and N_S introduces extra unwanted matching error. To account for minor discrepancies, we introduce a parameter $\gamma = 2$ as a tolerance threshold for errors. We then collect the number of cases where matches with larger error, specifically $\|N_S - N_{opt}\|_1 \geq \gamma$. This collection is performed at each 50 iterations throughout the distillation process, and the results are depicted in Fig. 2. Additionally, we present supplementary results with varying γ in Appendix A.1.

As observed in the figure, FTL methods consistently show a tendency to align steps with larger errors across different values of N_S. This behavior indicates that FTL methods compel trajectory matching with an inappropriate trajectory length. Consequently, student trajectories with an inappropriate length undergo either compression or stretching to conform to the diverse patterns exhibited by expert trajectories. This compels the synthetic dataset to overfit to the pre-buffered expert trajectories but sacrifices generality for unseen expert trajectories. We validate this observation with experimental results in Sec. 4.2, where AMP has been addressed. Additionally, the results obtained in Sec. 4.2 support that the challenge becomes particularly pronounced when synthetic data encounters unseen trajectories from unseen architectures.

AMP is Accumulative: We investigate into N_S by mimicing it as a small fluctuation δ. An additional term emerges in the prediction, denoted as $\theta_{N_S+\delta} = \theta_{N_S} + T_{D_S,f}(\theta_{N_S}, \delta)$. The presence of the additional term has implications for both the loss function, as depicted in Eq. (4), and the objectives outlined in Eq. (2) and Eq. (3). Consequently, this additional term accumulates with each update to the synthetic dataset D_S, a phenomenon we identify as accumulative.

AMP and Accumulated Trajectory Error: Previous research [16] revealed the existence of Accumulated Trajectory Error (ATE), which results from discrepancies between the initialization from previous student parameters and the assigned initial expert parameters in each iteration. Conversely, while AMP primarily focuses on the end of the trajectory for matching, it is noteworthy that AMP also contributes to ATE. In this subsection, we show the correlation between AMP and ATE.

As outlined in from [16] Sec. 3.1, the ATE is composed of the initialization error, the matching error ϵ_0, and the ATE from the previous iteration. The matching error ϵ_0, as part of ATE, is defined as

$$\epsilon_0 = T_{D_S,f}(\theta^*_{i,0}, N_S) - T_{D_T,f}(\theta^*_{i,0}, N_T). \tag{5}$$

$\theta^*_{i,0}$ is any network parameters from expert trajectories. We plug Eq. (1) in Eq. (5) and take L2 norm as distance measure. We define $e_t = \|\epsilon_0\|_2^2$, and AMP is indeed solving the following in each iteration as shown:

$$\min_t e_t = \min_{t \in \{0,1,\cdots,N_S\}} \|\theta'_{i,t} - \theta^*_{i,N_T}\|_2^2. \tag{6}$$

By addressing the AMP, we minimize matching errors at each step, thus iteratively reducing the ATE.

3.3 Automatic Training Trajectories (ATT)

The core idea behind ATT is to dynamically and adaptively select a suitable trajectory length based on the minimum distance policy. And ATT essentially

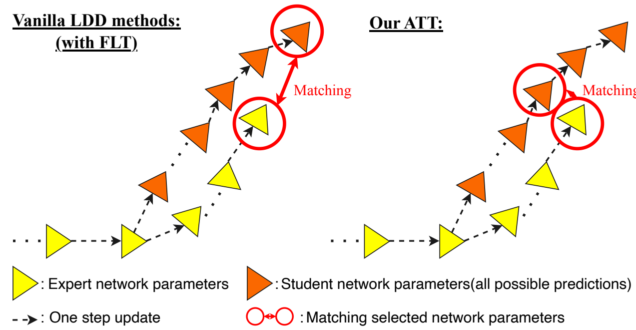

Fig. 3: The figure displays the core idea of our method ATT in comparison to traditional LDD methods employing FTL. Left: The Vanilla LDD bypasses all possible predictions and matches the inaccurate prediction. Right: our method ATT adopts an adaptive approach, aligning predictions with expert network parameters using a minimum distance policy. ATT avoid cases where compressing or stretching trajectories happen, and prevents the accumulation of errors resulting from those cases within the synthetic dataset at each iteration, thus achieves better distillation performances.

is a simple solution to Eq. (6). It computes all possible e_t and selects the t that gives the minimum e_t.

Fig. 3 provides an illustration of the ATT process. The process begins by unrolling a trajectory starting from the expert input $\theta^*_{i,0}$ on a synthetic dataset. Within each trajectory, every updated step $t \in \{0, 1, \cdots, N_S\}$ conveys a potential prediction $\theta'_{i,t}$. We evaluate the squared L2 distance between each prediction and the corresponding expert target with $e_i = \|\theta'_{i,t} - \theta^*_{i,N_T}\|^2_2$. The step N_{opt} with the minimum distance is selected out of all possible t, as depicted in Eq. (6). Subsequently, back-propagation is performed when matching at the selected prediction $\theta'_{i,N_{opt}}$.

We summarize pseudocode for ATT in Algorithm 1, which provides a step-by-step description of the proposed method. By employing this approach, we directly eliminate AMP, and thus reduce matching error. Further, given the significance of trajectory length in LDD methods, we conduct an ablation study in Sec. 4.4 to investigate its impact on ATT further.

4 Experiments

In this section, we present an overview of the dataset used, the experimental setups, and the baselines employed in our experiments in Sec. 4.1. Next, we showcase and compare the performance of our proposed method to the baseline approaches in cross-architecture performance Sec. 4.2, and on different datasets Sec. 4.3. Furthermore, we perform an ablation study on ATT and highlight its positive impact on boosting stability in Sec. 4.4. Lastly, we detail ATT's storage and computation requirement in Sec. 4.5.

Algorithm 1 Automatic Training Trajectories

```
Input: N_S: the number of training steps on D_S.
       N_M: the number of training steps on D_T.
       lr': learning rate for learning D_S
initialize lr', D_S
while not converged do
    randomly sample a pair of θ*_{i,0}, θ*_{i,N_M} from expert trajectories
    initialize θ'_{i,0} = θ*_{i,0}, d_t = [ ]
    for t = 0 to N_S do
        get θ'_{i,t+1} by a one-step update on θ'_{i,t}
        calculate e_t from Eq. (6) and append it to d_t
    end for
    find the index N_opt, which minimizes d_t
    conduct alternating optimization on Eq. (4) at early step N_opt over lr' and D_S
end while
Output: learning rate lr' and distilled data D_S
```

4.1 Dataset and Experimental Setup

CIFAR-10/100 [27]: The dataset comprises 50,000 training images and 10,000 test images, each with a size of 32x32 pixels. CIFAR-10/100 consists of 10 classes and 100 classes respectively. they are colored and natural.
64x64 Tiny ImageNet [13]: The dataset consists of 100,000 training images and 5,000 validation images, all with a size of 64x64 pixels. It encompasses 200 classes and and they are colored and natural.
ImageNet Subsets [13, 17]: We focus on subset ImageNette (assorted objects), ImageWoof (dog breeds), ImageFruit (fruits), ImageMeow (cats). Each of the subclasses has 10 classes of with the image size of 128x128.

Evaluation Setups: Following the methodologies established in previous works [6, 40, 58, 68, 69], we initiate the distillation process by generating a synthetic dataset using ATT, then assess its performance by training the synthetic dataset with five randomly initialized networks. The test accuracy of these networks is then measured on the original test dataset. To provide a comprehensive assessment, we calculate both the mean and variance of the test accuracy. To ensure a fair comparison, we adopt a simple ConvNet architecture, as designed by [19], following the precedent set by prior work [6, 58, 66, 69]. This ConvNet architecture incorporates convolutional layers [19], instance normalization [55], activation function RELU [1], and average pooling layers. The normalization layer remains unchanged in our experimental setup to maintain consistency and facilitate a direct comparison of methods.

Baselines: In our comparative analysis, we benchmark our research against the first DD method [59] and SDD methods including Dataset Condensation (DC) [69], Differentiable Siamese Augmentation (DSA) [66], Distribution matching (DM) [68], and Aligning Features (CAFE) [58]. Furthermore, we compare

DD methods with a meta-gradient computation-solving method: Kernel Inducing Points (KIP) [40]. Lastly, we include a comparison with LDD methods, encompassing the first LDD method Matching Training Trajectories (MTT) [6], Parameter Pruning (PP) [31], and Flat Trajectory Distillation (FTD) [16]. To facilitate a fair and meaningful comparison with different methods, our experimental setups deliberately exclude the incorporation of soft labels, network pruning, the adoption of batch normalization, or any additional factorization or parameterization methods. This intentional avoidance ensures that the comparison focuses on the core aspects of the compared methods without introducing confounding factors associated with the mentioned techniques. In this study, we refrain from comparing with instance selection approaches such as random selection, forgetting samples [52], or Herding [10]. This decision is based on the understanding that these methods have been significantly surpassed by dataset distillation in many prior works [6, 58, 66, 69]. Additionally, we do not include research that concentrates on soft-labels [5,11,51], data parameterization [14,25,34], data factorization [8,29,32,67], data and label encoding [14], selecting representative real samples [35], model augmentation [65], reduced budget for storage and transmissions with slimmable DD [34], approaches with only a few networks [33], or with model pool and label learning [74]. The exclusion of these methods is motivated by their orthogonality to our specific research focus, most of them can be implemented in conjunction with our methods as needed.

Evaluation Details on CIFAR-10/100: We adopt the simple 3-layer ConvNet architecture [19] ensure a justifiable comparison with other approaches. To maintain consistency with prior research [6, 40], we incorporate ZCA whitening and simple data augmentation techniques. Nonetheless, we did not employ Differentiable Siamese Augmentation (DSA) [66] in our baseline method to align identical setups with other baselines. Lastly, to ensure the justification and fairness of our comparison, we adopt a consistent test and learning setup aligned with other LDD methods [6, 16]. This choice aims at providing a fair platform for evaluating the effectiveness and performance of our approach in comparison to existing LDD methodologies.

Evaluate Details on Tiny ImageNet and ImageNet Subsets: Following previous MTT methods [6, 16], we extend our methodology to larger datasets. Specifically, we utilize the 4-layer ConvNet and 5-layer ConvNet as distillation models for Tiny ImageNet and ImageNet, respectively. To validate the effectiveness of our approach, we conduct experiments on six ImageNet subsets that were previously introduced to dataset distillation by [6]. This ensures that the evaluation is consistent and enables a meaningful assessment of the performance of our approach in comparison to existing methodologies.

4.2 Cross-Architecture Generalization

In the evaluation of DD, Cross-Architecture (CA) generalization serves as a crucial metric, gauging the synthetic dataset's capacity to be effectively learned by various architectures. Despite its significance, many prior works struggled

Table 1: The table presents the cross-architecture performance of CIFAR-10 with IPC=10. The network architectures listed in the table remain consistent with those used in previous works [6,16], with no modifications made to normalization or pooling layers to ensure fair performance comparisons. We reproduced MTT's result on IPC=50 and FTD's result on IPC=10 for a thorough assessment. The scores are all in percentage. Our method excels significantly in cross-architecture performance under various IPCs.

	ConvNet		ResNet18		VGG11		AlexNet	
IPC	10	50	10	50	10	50	10	50
KIP [40]	47.6 ± 0.9	-	36.8 ± 1.0	-	42.1 ± 0.4	-	24.4 ± 3.9	-
DSA [66]	52.1 ± 0.4	-	42.8 ± 1.0	-	43.2 ± 0.5	-	35.9 ± 1.3	-
MTT [6]	64.3 ± 0.7	71.6 ± 0.2	46.4 ± 0.6	61.9 ± 0.7	50.3 ± 0.8	55.4 ± 0.8	34.2 ± 2.6	48.2 ± 1.0
FTD [16]	66.1 ± 0.3	73.8 ± 0.2	53.2 ± 1.4	65.7 ± 0.3	47.0 ± 1.5	58.4 ± 1.6	35.9 ± 2.3	53.8 ± 0.9
ATT	$\mathbf{67.7 \pm 0.6}$	$\mathbf{74.5 \pm 0.4}$	$\mathbf{54.5 \pm 0.9}$	$\mathbf{66.3 \pm 1.1}$	$\mathbf{54.2 \pm 0.8}$	$\mathbf{61.7 \pm 0.9}$	$\mathbf{43.6 \pm 1.4}$	$\mathbf{60.0 \pm 0.9}$

to exhibit satisfactory generalization across different architectures. In this subsection, we delve into the examination of ATT in terms of cross-architecture generalization.

We evaluate the CA performance of ATT on ResNet18 [22], VGG11 [49], AlexNet [28], and the distillation architecture ConvNet, aligning with architectures studied in prior research. To ensure a fair comparison, we opt for simple network architectures to mitigate the impact of high test performance associated with more powerful structures. In line with closely related works, MTT [6] and FTD [16], which assessed CA performance on different IPCs, we present CA results for our methods in both scenarios. Additionally, we replicate their missing CA evaluations on the other IPC, contributing to a comprehensive and balanced assessment.

The results, as presented in Tab. 1, showcase the remarkable performance of our proposed method. ATT significantly outperforms previous methods in CA generalization. Specifically, compared to the previous best result with IPC=10, ATT achieves an improvement of 1.3% on ResNet18, 7.2% on VGG11, and 7.7% on AlexNet. Furthermore, for IPC=50, ATT demonstrates improvements of 3.3% on VGG11 and 6.2% on AlexNet.

This substantial improvement can be attributed to the mitigation of AMP, as discussed in Sec. 3.2. FTL methods, as revealed in previous discussions, tend to force student trajectories to match with an inappropriate trajectory length, leading to AMP. Consequently, FTL methods exhibit overfitting to seen expert trajectories while sacrificing generalization to unseen trajectories. The CA performance of ATT underscores the significance of addressing the AMP, especially when dealing with unseen trajectories from previously unencountered architectures.

4.3 Benchmark Comparison

Evaluation of distilled architecture performance serves as a crucial metric in dataset distillation, gauging the efficacy of synthetic dataset learning on refined architectures. Our test performance results are detailed in Tab. 2 and Tab. 3.

Table 2: The table showcases performance comparison to benchmark methods in terms of test accuracy. LD and DD use AlexNet for CIFAR-10, and all the rest are DD methods using the simple ConvNet for the distillation process. IPC: Images Per Class. The numbers in table are presented in percentage. As shown, our method generally outperforms existing methods on various dataset and IPCs.

	CIFAR-10			CIFAR-100			Tiny ImageNet	
Full dataset	84.8 ± 0.1			56.2 ± 0.3			37.6 ± 0.4	
IPC	1	10	50	1	10	50	1	10
DD [59]	-	36.8 ± 1.2	-	-	-	-	-	-
DC [69]	28.3 ± 0.5	44.9 ± 0.5	53.9 ± 0.5	12.8 ± 0.3	25.2 ± 0.3	-	-	-
DSA [66]	28.8 ± 0.7	52.1 ± 0.5	60.6 ± 0.5	13.9 ± 0.3	32.3 ± 0.3	42.8 ± 0.4	-	-
DM [68]	26.0 ± 0.8	48.9 ± 0.6	63.0 ± 0.4	11.4 ± 0.3	29.7 ± 0.3	43.6 ± 0.4	3.9 ± 0.2	12.9 ± 0.4
CAFE [58]	30.0 ± 1.1	46.3 ± 0.6	55.5 ± 0.6	12.9 ± 0.3	27.8 ± 0.3	37.9 ± 0.3	-	-
KIP [40]	$\mathbf{49.9 \pm 0.2}$	62.7 ± 0.3	68.6 ± 0.2	15.7 ± 0.2	28.3 ± 0.1	-	-	-
MTT [6]	46.2 ± 0.8	65.4 ± 0.7	71.6 ± 0.2	24.3 ± 0.3	39.7 ± 0.4	47.7 ± 0.2	8.8 ± 0.3	23.2 ± 0.2
PP [31]	46.4 ± 0.6	65.5 ± 0.3	71.9 ± 0.2	24.6 ± 0.1	43.1 ± 0.3	48.4 ± 0.3	-	-
FTD [16]	46.8 ± 0.3	66.6 ± 0.3	73.8 ± 0.2	25.2 ± 0.2	43.4 ± 0.3	50.7 ± 0.3	10.4 ± 0.3	24.5 ± 0.2
ATT	$\mathbf{48.3 \pm 1}$	$\mathbf{67.7 \pm 0.6}$	$\mathbf{74.5 \pm 0.4}$	$\mathbf{26.1 \pm 0.3}$	$\mathbf{44.2 \pm 0.5}$	$\mathbf{51.2 \pm 0.3}$	$\mathbf{11.0 \pm 0.5}$	$\mathbf{25.8 \pm 0.4}$

ATT, addressing the AMP, aims at improving generality on trajectories not encountered before. Thus, the result additionally reflect how much they suffer from AMP.

Notably, on datasets with substantial image sizes, such as ImageNet with IPC=10, ATT exhibits a significant improvement over the previous best method: 2.2% on subset Meow, 1.6% on subset Nette, 1.2% on subset Woof, and 1% on subset Fruit. The impact extends to datasets with increased class complexity, exemplified on Tiny ImageNet, where improvements of 1.3% on IPC=10 and 0.6% on IPC=1 are observed. CIFAR-10 experiences notable enhancements as well, with improvements of 1.5% on IPC=1 over FTD, 1.1% on IPC=10 over previous best result. CIFAR-100 also sees improvements, with up to 0.9% at different IPCs. While KIP demonstrate commendable performance in IPC=1 on the CIFAR-10 dataset, ATT significantly outperforms them on larger IPCs and more extensive datasets.

It is crucial to note that, while test performance on distilled architecture is generally significant, it is not as pronounced as those on unseen networks from unfamiliar architectures, as discussed in Sec. 4.2. This disparity arises from FTL methods potentially overfitting to pre-buffered expert trajectories, which does not include other network architectures other than the distilled architecture.

4.4 Ablation Study on Parameters

Trajectory Bounds N_S: In order to accommodate diverse memory constraints, ATT incorporates the trajectory length parameter N_S to examine the scope of the search for the optimal length. In our experimental analysis, we delve into the behavior of ATT, specifically examining the step selection process during distillation. This investigation focuses on delineating how the optimal step N_{opt} evolves under different trajectory length constraints throughout the distillation process.

Table 3: Performance comparison of our method ATT to other baselines on ImageNet subset. All the scores are presented in percentage. As shown, our method demonstrates significant improvement on large dataset ImageNet, especially on IPC=10

		ImageNet Subsets			
Methods	IPC	ImageNette	ImageWoof	ImageFruit	ImageMeow
Full Dataset		87.4 ± 1.0	67.0 ± 1.3	63.9 ± 2.0	66.7 ± 1.1
MTT [6]	1	47.7 ± 0.9	28.6 ± 0.8	26.6 ± 0.8	30.7 ± 1.6
	10	63.0 ± 1.3	35.8 ± 1.8	40.3 ± 1.3	40.4 ± 2.2
FTD [16]	1	52.2 ± 1.0	30.1 ± 1.0	29.1 ± 0.9	33.8 ± 1.5
	10	67.7 ± 0.7	38.8 ± 1.4	44.9 ± 1.5	43.3 ± 0.6
ATT	1	$\mathbf{52.4 \pm 1.1}$	$\mathbf{31.2 \pm 1.8}$	$\mathbf{30.0 \pm 0.8}$	$\mathbf{34.0 \pm 1.4}$
	10	$\mathbf{69.3 \pm 0.8}$	$\mathbf{40.0 \pm 1.4}$	$\mathbf{45.9 \pm 1.2}$	$\mathbf{45.5 \pm 1.6}$

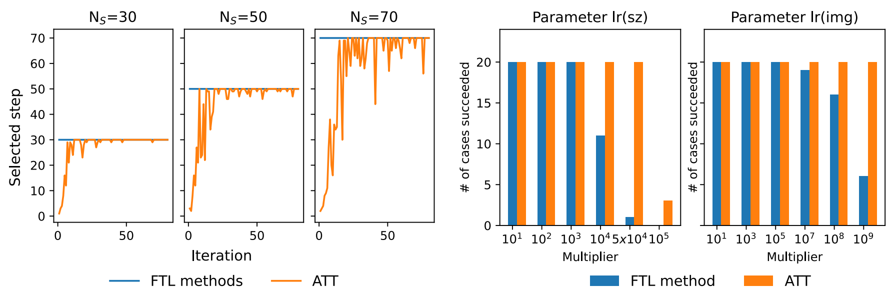

Fig. 4: Left: This plot illustrates ATT's step selection during the distillation phase, highlighting its preference for smaller steps initially, contributing to enhanced stability during parameter tuning. Right: The plot demonstrates ATT's superior stability in varying parameters, showcasing the number of successful cases when alter parameter to mutiplier times. Different multipliers are presented as distinct cases.

Our experiments are conducted on CIFAR-10 with IPC=1, and we emperically choose N_S to be 30, 50, 70, indicating cases with not sufficient steps, close to optimal steps, and too much steps. The results, as depicted on left of Fig. 4, highlight the distinction between FTL and ATT. While FTL consistently opts for a fixed trajectory length, ATT tends to dynamically adjust its trajectory length, starting with smaller values at the beginning and subsequently adapting, fluctuating below N_S to minimize errors. The behavior remains with different N_S, and we will show more results on various N_S and Iterations in Appendix A.2. This adaptive behavior allows ATT to perform short trajectory matching initially instead of enforcing larger trajectory matching.

ATT and Parameter Variations: We empirically substantiate ATT's stability toward parameter variation by introducing variations in two sensitive parameters: the learning rate of pixels, denoted as lr(img) and the learning rate of step size, denoted as lr(sz). They are tuning parameters that determine the step size for the synthetic dataset and the trainable learning rate, respectively. We align the initial parameters with the vanilla LDD method, and multiply the examining parameters by factors of multiplier. Specifically, we do not make additional changes to variables other than the one under examination. Each experiment is repeated twenty times, and we document the number of successful cases under each parameter setup. Cases of failure are identified by divergent loss or negative trainable learning rate, while cases with convergent loss are considered successful.

As demonstrated on the right of Fig. 4, ATT exhibits higher stability under different parameter sets. This is attributed to ATT allowing and selecting a small step size for the student trajectory, facilitating the synthetic data to learn from early trajectories at the beginning, instead of enforcing larger trajectory matching. This advantage holds significant support when undertaking DD on a new dataset.

4.5 Storage and Computation

Similar to other LDD methods such as [6,16], computational overhead and storage requirements present challenges. However, LDD methods yield synthetic datasets with superior test performances across various network architectures, benefiting end-users at the cost of increased resources for producers. In our experiments, we primarily use five A100 GPUs, and utilize 100 expert trajectories as the default setting. Assessing the storage requirements for expert trajectories, we find approximately 60MB for each CIFAR expert, and 120MB for each ImageNet expert. We list more details in Appendix A.5.

While ATT incurs time in employing distance measurement with L2 norm, it strategically prioritizes matching steps where $t < N_S$, enabling early backpropagation on trajectories and avoiding unnecessary computations at the tail end. This approach serves as a time-efficient compensation for the distance calculation. To illustrate, we conduct experiments on CIFAR-10 with IPC=1, comparing our method to the vanilla LDD approach. The experiments employ identical parameters and hardware setups, repeated ten times, and the average time per iteration is determined. As a result, both the vanilla LDD method and ATT exhibit a similar runtime of 0.8s per iteration.

5 Conclusions

In this paper, we delved into long-range matching dataset distillation methods, and reveal that traditional long-range matching methods exhibit persistent limitations and mismatches in aligning trajectories, resulting in the Accumulated Mismatching Problem (AMP). We further reveal AMP's contribution to matching error, and propose ATT to address AMP. Empirical results showcase that ATT not only rectifies the AMP, but also enhances stability when confronted with parameter changes, and the method exhibits superior cross-architecture performance.

As we continue to explore innovative methodologies in dataset distillation, the insights gained from this research pave the way for more effective and adaptable approaches in the field.

Acknowledgment This work is funded by Bayerische Forschungsstiftung under the research grants *Von der Edge zur Cloud und zurück: Skalierbare und Adaptive Sensordatenverarbeitung* (AZ-1468-20), and supported by AI systems hosted and operated by the Leibniz-Rechenzentrum (LRZ) der Bayerischen Akademie der Wissenschaften. Further, part of the results have been obtained on systems in the test environment BEAST (Bavarian Energy Architecture & Software Testbed) at the Leibniz Supercomputing Centre.

References

1. Agarap, A.F.: Deep learning using rectified linear units (relu). arXiv preprint arXiv:1803.08375 (2018)
2. Aljundi, R., Lin, M., Goujaud, B., Bengio, Y.: Gradient based sample selection for online continual learning. Advances in neural information processing systems **32** (2019)
3. Assadi, S., Bateni, M., Bernstein, A., Mirrokni, V., Stein, C.: Coresets meet edcs: algorithms for matching and vertex cover on massive graphs. In: Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 1616–1635. SIAM (2019)
4. Bachem, O., Lucic, M., Krause, A.: Practical coreset constructions for machine learning. arXiv preprint arXiv:1703.06476 (2017)
5. Bohdal, O., Yang, Y., Hospedales, T.: Flexible dataset distillation: Learn labels instead of images. arXiv preprint arXiv:2006.08572 (2020)
6. Cazenavette, G., Wang, T., Torralba, A., Efros, A.A., Zhu, J.Y.: Dataset distillation by matching training trajectories. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4750–4759 (2022)
7. Cazenavette, G., Wang, T., Torralba, A., Efros, A.A., Zhu, J.Y.: Wearable imagenet: Synthesizing tileable textures via dataset distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2278–2282 (2022)
8. Cazenavette, G., Wang, T., Torralba, A., Efros, A.A., Zhu, J.Y.: Generalizing dataset distillation via deep generative prior. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3739–3748 (2023)
9. Chen, Y., Welling, M.: Parametric herding. In: Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. pp. 97–104. JMLR Workshop and Conference Proceedings (2010)
10. Chen, Y., Welling, M., Smola, A.: Super-samples from kernel herding. arXiv preprint arXiv:1203.3472 (2012)
11. Cui, J., Wang, R., Si, S., Hsieh, C.J.: Scaling up dataset distillation to imagenet-1k with constant memory (2022)
12. Dasgupta, A., Drineas, P., Harb, B., Kumar, R., Mahoney, M.W.: Sampling algorithms and coresets for \ell_p regression. SIAM Journal on Computing **38**(5), 2060–2078 (2009)
13. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). https://doi.org/10.1109/CVPR.2009.5206848
14. Deng, Z., Russakovsky, O.: Remember the past: Distilling datasets into addressable memories for neural networks (2022)
15. Dong, T., Zhao, B., Lyu, L.: Privacy for free: How does dataset condensation help privacy? In: International Conference on Machine Learning. pp. 5378–5396. PMLR (2022)
16. Du, J., Jiang, Y., Tan, V.T., Zhou, J.T., Li, H.: Minimizing the accumulated trajectory error to improve dataset distillation. arXiv preprint arXiv:2211.11004 (2023)
17. Fastai: A smaller subset of 10 easily clas- sified classes from imagenet, and a little more french
18. Feldman, D.: Core-sets: Updated survey. Sampling techniques for supervised or unsupervised tasks pp. 23–44 (2020)

19. Gidaris, S., Komodakis, N.: Dynamic few-shot visual learning without forgetting. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4367–4375 (2018)
20. 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)
21. Har-Peled, S., Kushal, A.: Smaller coresets for k-median and k-means clustering. In: Proceedings of the twenty-first annual symposium on Computational geometry. pp. 126–134 (2005)
22. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
23. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
24. Kendall, A., Gal, Y.: What uncertainties do we need in bayesian deep learning for computer vision? Advances in neural information processing systems **30** (2017)
25. 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: International Conference on Machine Learning. pp. 11102–11118. PMLR (2022)
26. Kiyasseh, D., Zhu, T., Clifton, D.A.: Pcps: Patient cardiac prototypes to probe ai-based medical diagnoses, distill datasets, and retrieve patients. Transactions on Machine Learning Research (2022)
27. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
28. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems **25** (2012)
29. Lee, H.B., Lee, D.B., Hwang, S.J.: Dataset condensation with latent space knowledge factorization and sharing. arXiv preprint arXiv:2208.10494 (2022)
30. Lee, S., Chun, S., Jung, S., Yun, S., Yoon, S.: Dataset condensation with contrastive signals. In: International Conference on Machine Learning. pp. 12352–12364. PMLR (2022)
31. Li, G., Togo, R., Ogawa, T., Haseyama, M.: Dataset distillation using parameter pruning. arXiv preprint arXiv:2209.14609 (2022)
32. Liu, S., Wang, K., Yang, X., Ye, J., Wang, X.: Dataset distillation via factorization (2022)
33. Liu, S., Wang, X.: Few-shot dataset distillation via translative pre-training. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 18654–18664 (October 2023)
34. Liu, S., Ye, J., Yu, R., Wang, X.: Slimmable dataset condensation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3759–3768 (2023)
35. Liu, Y., Gu, J., Wang, K., Zhu, Z., Jiang, W., You, Y.: Dream: Efficient dataset distillation by representative matching (2023)
36. Liu, Y., Li, Z., Backes, M., Shen, Y., Zhang, Y.: Backdoor attacks against dataset distillation. arXiv preprint arXiv:2301.01197 (2023)
37. Loo, N., Hasani, R., Lechner, M., Rus, D.: Dataset distillation fixes dataset reconstruction attacks. arXiv preprint arXiv:2302.01428 (2023)
38. Loo, N., Hasani, R., Lechner, M., Rus, D.: Dataset distillation with convexified implicit gradients (2023)

39. Mirrokni, V., Zadimoghaddam, M.: Randomized composable core-sets for distributed submodular maximization. In: Proceedings of the forty-seventh annual ACM symposium on Theory of computing. pp. 153–162 (2015)
40. Nguyen, T., Novak, R., Xiao, L., Lee, J.: Dataset distillation with infinitely wide convolutional networks. Advances in Neural Information Processing Systems **34**, 5186–5198 (2021)
41. Nickolls, J., Dally, W.J.: The gpu computing era. IEEE micro **30**(2), 56–69 (2010)
42. Olvera-López, J.A., Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F., Kittler, J.: A review of instance selection methods. Artificial Intelligence Review **34**, 133–143 (2010)
43. O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Hernandez, G.V., Krpalkova, L., Riordan, D., Walsh, J.: Deep learning vs. traditional computer vision. In: Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1. pp. 128–144. Springer (2020)
44. Paul, M., Ganguli, S., Dziugaite, G.K.: Deep learning on a data diet: Finding important examples early in training. Advances in Neural Information Processing Systems **34**, 20596–20607 (2021)
45. Pi, R., Zhang, W., Xie, Y., Gao, J., Wang, X., Kim, S., Chen, Q.: Dynafed: Tackling client data heterogeneity with global dynamics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12177–12186 (2023)
46. Sangermano, M., Carta, A., Cossu, A., Bacciu, D.: Sample condensation in online continual learning. In: 2022 International Joint Conference on Neural Networks (IJCNN). pp. 01–08. IEEE (2022)
47. Schwartz, R., Dodge, J., Smith, N.A., Etzioni, O.: Green ai. Communications of the ACM **63**(12), 54–63 (2020)
48. Sener, O., Savarese, S.: Active learning for convolutional neural networks: A coreset approach. arXiv preprint arXiv:1708.00489 (2017)
49. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
50. Song, R., Liu, D., Chen, D.Z., Festag, A., Trinitis, C., Schulz, M., Knoll, A.: Federated learning via decentralized dataset distillation in resource-constrained edge environments. arXiv preprint arXiv:2208.11311 (2022)
51. Sucholutsky, I., Schonlau, M.: Soft-label dataset distillation and text dataset distillation. In: 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2021)
52. Toneva, M., Sordoni, A., Combes, R.T.d., Trischler, A., Bengio, Y., Gordon, G.J.: An empirical study of example forgetting during deep neural network learning. arXiv preprint arXiv:1812.05159 (2018)
53. Tsang, I.W., Kwok, J.T., Cheung, P.M., Cristianini, N.: Core vector machines: Fast svm training on very large data sets. Journal of Machine Learning Research **6**(4) (2005)
54. Tukan, M., Maalouf, A., Feldman, D.: Coresets for near-convex functions. Advances in Neural Information Processing Systems **33**, 997–1009 (2020)
55. Ulyanov, D., Vedaldi, A., Lempitsky, V.: Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022 (2016)
56. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E., et al.: Deep learning for computer vision: A brief review. Computational intelligence and neuroscience **2018** (2018)
57. Wang, J., Guo, S., Xie, X., Qi, H.: Protect privacy from gradient leakage attack in federated learning. In: IEEE INFOCOM 2022-IEEE Conference on Computer Communications. pp. 580–589. IEEE (2022)

58. 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: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12196–12205 (2022)
59. Wang, T., Zhu, J.Y., Torralba, A., Efros, A.A.: Dataset distillation. arXiv preprint arXiv:1811.10959 (2018)
60. Wiewel, F., Yang, B.: Condensed composite memory continual learning. In: 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2021)
61. Xu, Z., Chen, Y., Pan, M., Chen, H., Das, M., Yang, H., Tong, H.: Kernel ridge regression-based graph dataset distillation. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. pp. 2850–2861 (2023)
62. Yang, S., Xie, Z., Peng, H., Xu, M., Sun, M., Li, P.: Dataset pruning: Reducing training data by examining generalization influence. arXiv preprint arXiv:2205.09329 (2022)
63. Yin, Z., Xing, E., Shen, Z.: Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. Advances in Neural Information Processing Systems **36** (2024)
64. Yoon, J., Madaan, D., Yang, E., Hwang, S.J.: Online coreset selection for rehearsal-based continual learning. arXiv preprint arXiv:2106.01085 (2021)
65. Zhang, L., Zhang, J., Lei, B., Mukherjee, S., Pan, X., Zhao, B., Ding, C., Li, Y., Xu, D.: Accelerating dataset distillation via model augmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11950–11959 (2023)
66. Zhao, B., Bilen, H.: Dataset condensation with differentiable siamese augmentation. In: International Conference on Machine Learning. pp. 12674–12685. PMLR (2021)
67. Zhao, B., Bilen, H.: Synthesizing informative training samples with gan. arXiv preprint arXiv:2204.07513 (2022)
68. Zhao, B., Bilen, H.: Dataset condensation with distribution matching. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 6514–6523 (2023)
69. Zhao, B., Mopuri, K.R., Bilen, H.: Dataset condensation with gradient matching. arXiv preprint arXiv:2006.05929 (2020)
70. Zhao, G., Li, G., Qin, Y., Yu, Y.: Improved distribution matching for dataset condensation (2023)
71. Zhmoginov, A., Sandler, M., Miller, N., Kristiansen, G., Vladymyrov, M.: Decentralized learning with multi-headed distillation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8053–8063 (2023)
72. Zhou, Y., Ma, X., Wu, D., Li, X.: Communication-efficient and attack-resistant federated edge learning with dataset distillation. IEEE Transactions on Cloud Computing (2022)
73. Zhou, Y., Pu, G., Ma, X., Li, X., Wu, D.: Distilled one-shot federated learning. arXiv preprint arXiv:2009.07999 (2020)
74. Zhou, Y., Nezhadarya, E., Ba, J.: Dataset distillation using neural feature regression. arXiv preprint arXiv:2206.00719 (2022)