DATA-EVOLUTION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advancements in machine learning have been driven by models trained on large-scale, high-quality datasets. However, the practical application of these models faces two significant challenges: the infeasibility of acquiring precise labels in real-world settings and the substantial computational burden imposed by training large models. While existing approaches—such as self-supervised learning, weak supervision, noisy label learning, and dataset distillation—address these challenges from a model-centric perspective, they often overlook the potential benefits of optimizing the data itself. This paper introduces a novel data-centric learning paradigm where both the dataset and the model co-evolve during the learning process. We formalize this paradigm and propose a Data-evolution Learning Algorithm (DELA), which offers three key advantages: optimized dataset generation, versatile dataset compatibility, and effective utilization of prior knowledge. Extensive experiments demonstrate that DELA enables the creation of optimized datasets for reuse in subsequent training, effectively addressing diverse datasets with varying target types. Moreover, DELA accelerates learning by utilizing architecture-agnostic, open-source prior models for efficient data creation. Notably, DELA frequently outperforms traditional SOTA model-centric methods in self-supervised and noisy label learning. Furthermore, its simplicity enables implementation in only *two lines* of PyTorch code, offering significant potential for advancements in representation learning. Our code will be made publicly available.

1 INTRODUCTION

Machine learning models have demonstrated exceptional performance across a broad spectrum of applications, as exemplified by GPT-4 (Achiam et al., 2023) and LVM (Bai et al., 2023). The success of these models is largely due to their remarkable representational capabilities, which are typically achieved through training on large-scale, high-quality datasets with comprehensive and accurate supervision (Deng et al., 2009; Radford et al., 2021). However, the proliferation of massive datasets in contemporary deep learning presents two critical challenges: (i) *Acquiring such precise labels in the real world is often infeasible due to several factors, including the high cost of annotation (Settles et al., 2008; Gadre et al., 2023), the inherent biases and subjectivity of annotators (Tommasi et al., 2017; Pagano et al., 2023), and privacy concerns (Mireshghallah et al., 2020; Strobel & Shokri, 2022).* (ii) *Training large models with increasing data and model capacity imposes a huge computational burden (Brown et al., 2020; Cheng et al., 2017; Strubell et al., 2019).*

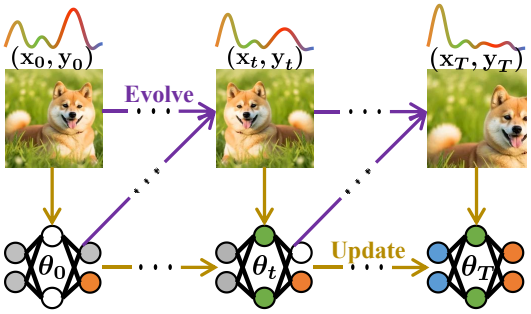


Figure 1: This panel visually summarizes the Data-Evolution Learning paradigm introduced in our study. This paradigm is designed to: (1) gradually evolve both the data and the model throughout the learning process, and (2) accept various types of data as input, including targets generated by a randomly initialized model and human annotation. This method facilitates efficient model training and data evolution across heterogeneous datasets within a unified framework. The data, once evolved, can be stored, thereby improving the effectiveness and efficiency of future training processes.

The community has made significant strides in addressing challenges (i) and (ii) through various approaches: self-supervised learning (Chen et al., 2020a; Caron et al., 2021; Chen & He, 2021; Bardes et al., 2021), weak supervision learning (Zhou, 2018; Sugiyama et al., 2022; Chen et al.,

2024), noisy label learning (Xu et al., 2019; Wang et al., 2024b; 2019; Han et al., 2020b), and dataset distillation (Wang et al., 2018; Sun et al., 2024; Shao et al., 2023). However, these model-centric approaches primarily concentrate on developing methods that facilitate effective learning from defective data without altering the data itself. In this paper, we propose to address challenges (i) and (ii) from a data-centric perspective by concurrently evolving both the initialized data and the model into more optimal forms. This approach allows for the dynamic modification of both data and the model throughout the data-evolution learning process.

To achieve this goal, we formally define our objective in Definition 1 and propose DELA that offers three distinct advantages over model-centric learning approaches: (a) *Optimized Dataset Generation*: Upon completion of the learning process, an optimized dataset is produced, which can be utilized to train another model with improved efficiency. (b) *Versatile Dataset Compatibility*: DELA is compatible with datasets containing various target types, including noisy labels and unlabeled data, in contrast to model-centric methods that are typically tailored for specific data types. (c) *Effective Utilization of Prior Knowledge*: In scenarios involving unlabeled data, DELA facilitates the integration of prior knowledge from open-source models for initializing data. This approach accelerates the learning process and conserves computational resources.

Definition 1 (A learning paradigm of simultaneously evolving datasets and models) . *Given a dataset $D = (D_X, D_Y)$ and model $\phi_{\boldsymbol{\theta}}$, our objective is to derive an evolved dataset D' and model $\phi_{\boldsymbol{\theta}_D}$ after multiple iterations of a data-evolution algorithm, ensuring:*

$$\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P}\left[\ell(\phi_{\boldsymbol{\theta}_{D'}}(\mathbf{x}),\mathbf{y})\right] < \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim P}\left[\ell(\phi_{\boldsymbol{\theta}_D}(\mathbf{x}),\mathbf{y})\right] < \epsilon\,, \tag{1}$$

where P is the test real distribution, $\mathbf{x}$ is a data sample, ℓ is the loss function, and ϵ is a predefined small threshold. Here, $\boldsymbol{\theta}_{D'}$ denotes the parameters of the neural network ϕ trained on D'. Furthermore, the training on D' should use the same or a smaller training budget compared to D.

Our work presents the following **five key contributions below** as an initial step toward establishing a data-centric representation learning paradigm:

(a) *Defining Data-Evolution Learning (see Definition 1) and Figure 1*. To the best of our knowledge, we are the first to introduce a data-centric learning paradigm where both data and model co-evolve simultaneously, and we formalize the objective of this task. Additionally, we present a simple and effective technical framework, DELA, to achieve this objective, highlighting several advantages of this paradigm that remain intractable for traditional model-centric learning paradigms.

(b) *An optimized dataset is obtained through learning using* DELA *(see Section 4.2).* Traditional model-centric approaches focus on training an effective model on a fixed dataset, where the dataset remains unchanged after the learning process. This paradigm typically results in each training procedure being independent, meaning subsequent training processes cannot leverage information from previous sessions, especially when involving models with different architectures. DELA addresses this limitation by evolving the dataset during training, allowing it to be reused and further optimized in subsequent training processes involving models with varying architectures.

(c) *The learning process of* DELA *can be significantly accelerated (see Section 4.3).* Leveraging open-source pre-trained models as prior knowledge is a widely adopted approach in contemporary machine learning to expedite and improve task-specific training. For example, Ridnik et al. (2021) use a ResNet-50 model pre-trained on ImageNet-1K as a starting point to train on ImageNet-21K. However, this approach may not be effective when the target model is based on a different architecture. In contrast, DELA can fully exploit prior knowledge from architecture-agnostic models to accelerate learning by employing data initialization (see Section 3).

(d) *Addressing diverse datasets constructed with varying types of targets (see Section 4.4).* Specifically, unlike previous approaches such as TNLPAD (Wang et al., 2024b) and SimCLR (Chen et al., 2020a), which are tailored for noise-labeled and unlabeled data respectively, DELA leverages multiple data initializations across different datasets. This strategy enables simultaneous model training and dynamic data evolution across various dataset types.

(e) DELA *demonstrates competitive model training performance in comparison to several model-centric learning approaches (see Section 4.5).* Extensive experiments conducted on four widely-used datasets, involving three neural network architectures and seven model-centric learning algorithms,

validate the effectiveness and efficiency of DELA. Notably, DELA matches or outperforms certain state-of-the-art (SOTA) model-centric methods.

2 RELATED WORK

This section integrates three key areas of deep learning: (a) techniques for dataset condensation that maintain efficacy; (b) self-supervised learning methods for training models on unlabeled data; (c) noise-robust learning approaches for training models on noisy data.

2.1 DATASET DISTILLATION: EFFICIENT AND EFFECTIVE LEARNING WITH REDUCED DATA

The goal of dataset distillation is to generate a substantially smaller dataset that maintains performance comparable to the original dataset. Traditional methods replicate behaviors from the original dataset within the distilled one, aiming to reduce discrepancies between surrogate neural network models trained on both synthetic and original datasets. Key metrics in this process include gradient matching (Zhao et al., 2020; Kim et al., 2022; Zhang et al., 2023; Liu et al., 2023), feature alignment (Wang et al., 2022), distribution matching (Zhao & Bilen, 2023; Zhao et al., 2023), and training trajectory consistency (Cazenavette et al., 2022; Cui et al., 2022; Du et al., 2023; Cui et al., 2023; Yu et al., 2023; Guo et al., 2023). However, these approaches entail significant computational overhead due to the continuous calculation of discrepancies between the distilled and original datasets. The optimization process requires multiple iterations to minimize these discrepancies until convergence, making it challenging to scale to large datasets, such as ImageNet (Deng et al., 2009).

A promising strategy involves developing metrics that capture essential dataset information, enabling efficient scaling to large datasets like ImageNet-1K with larger backbones. This approach avoids the need for multiple comparisons between original and distilled datasets. For instance, SRe^{2}L (Yin et al., 2023) condenses an entire dataset into a model using pre-trained neural networks, such as ResNet-18 (He et al., 2016), and subsequently extracts knowledge from these models into images and targets, thereby forming a distilled dataset. Recently, RDED (Sun et al., 2024) suggests that images accurately recognized by competent observers, such as humans and pre-trained models, are more valuable for learning.

2.2 SELF-SUPERVISED LEARNING: EXTRACTING REPRESENTATION FROM UNLABELED DATA

The primary objective of self-supervised learning is to generate robust representations independent of human-labeled data. These representations should rival those obtained through supervised learning, offering robust performance across a range of tasks.

Contrastive learning-based methods implicitly assign a "one-hot" label to each sample and its augmented versions to enhance discriminative power. Since the introduction of InfoNCE (Oord et al., 2018), numerous studies (He et al., 2020; Chen et al., 2020a;b; Cao et al., 2020; Kalantidis et al., 2020; Chen et al., 2021; Zhu et al., 2021; Li et al., 2021; Caron et al., 2020; Hu et al., 2021) have advanced this area. MoCo (He et al., 2020; Chen et al., 2020b; 2021) utilizes a momentum encoder to maintain consistent negatives, proving effective for both CNNs and Vision Transformers. SimCLR (Chen et al., 2020a) incorporates strong data augmentations and a nonlinear projection head. Other approaches include instance classification (Cao et al., 2020), enhanced data augmentation (Kalantidis et al., 2020; Zhu et al., 2021), clustering techniques (Li et al., 2021; Caron et al., 2020), and adversarial training (Hu et al., 2021). These methods improve the alignment and uniformity of representations on the hypersphere (Wang & Isola, 2020).

Asymmetric network methods facilitate self-supervised learning by leveraging only positive pairs, effectively avoiding representational collapse through the use of asymmetric architectures. BYOL (Jean-Bastien et al., 2020) employs a dual-component framework consisting of a predictor network and a momentum encoder. Richemond et al. (2020) demonstrate that BYOL performs efficiently without the need for batch statistics. SimSiam (Chen & He, 2021) implements gradient stopping on the target branch to replicate the function of momentum encoder. DINO (Caron et al., 2021) incorporates a self-distillation loss mechanism. UniGrad (Tao et al., 2022) combines asymmetric networks with contrastive learning techniques within a theoretically cohesive framework.

In addition, Singh et al. (2023) explore a straightforward self-training technique aimed at improving supervised fine-tuning performance, which can be viewed as applying expectation-maximization for reinforcement learning.

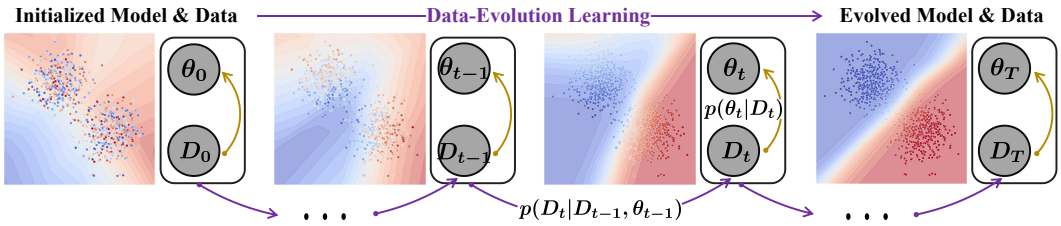


Figure 2: **This panel intuitively illustrates the Data-Evolution Learning framework**: (a) During the training process over T steps, our algorithm updates the parameter $\boldsymbol{\theta}_{t-1}$ using the dataset D_{t-1} at time $t-1$. (b) Subsequently, the dataset D_{t-1} evolves into D_t based on the model, now with updated parameters $\boldsymbol{\theta}_{t-1}$, at time t. Together, (a) and (b) constitute a loop in the data-model co-evolution learning framework.

2.3 Noisy Label Learning: Training Models Over Noise-Labeled Datasets

Learning with noisy labels has garnered significant attention in recent years (Han et al., 2020b; Yao et al., 2018). Numerous approaches have been developed to address this challenge, generally dividing into two categories: those leveraging the memorization effect and those that do not.

Existing noisy label learning methods can be roughly grouped into two categories: (a) Employing an explicit or implicit noise model to estimate the distributions of noisy and clean labels, subsequently deleting or correcting the noisy samples. The models used can vary, including neural networks (Goldberger & Ben-Reuven, 2017; Jiang et al., 2018; Lee et al., 2018; Ren et al., 2018), conditional random fields (Vahdat, 2017), or knowledge graphs (Li et al., 2017). However, a significant limitation is their reliance on a substantial number of clean training samples, making them unsuitable for many datasets dominated by noisy labels. (a) Constructing more balanced loss functions to mitigate the impact of noisy training samples (Liu & Tao, 2015; Ma et al., 2018; Zhang & Sabuncu, 2018; Wang et al., 2019; Xu et al., 2019). For instance, TNLPAD (Wang et al., 2024b) addresses noisy labels through network parameter additive decomposition.

3 A Simple Framework for Data-Evolution Learning

This study introduces a novel approach for data-evolution learning (DELA), focusing on evolving data during training (see Figure 2). The central concept of this method is to facilitate the "collaboration" between data and models. Specifically, the methodology is systematically divided into multiple stages: (a) Initialize targets for given samples; (b) Update the model for one step using the current data; (c) Evolve the data using the updated model, and then revert to step (b).

(a) Initializing targets for given samples. We hypothesize that the primary distinction among datasets, such as CIFAR-10 and its noise-labeled variant, is primarily in their targets. The samples themselves, sourced consistently from the real world, exhibit relatively similar. Within this framework, both samples and targets are treated as variables during training, necessitating the use of scenario-specific strategies for target initialization. Formally, we define:

$$D \leftarrow \{(\mathbf{x}, \mathbf{y}) \mid \mathbf{y} = \boldsymbol{\psi}(\mathbf{x}),\, \mathbf{x} \in D_X\}\,, \tag{2}$$

where $\boldsymbol{\psi}$ denotes the "data-initializer". For instance, $\boldsymbol{\psi}$ can represent human annotations and $\mathbf{y}$ indicates the corresponding one-hot labels for standard dataset like CIFAR-10. Additionally, $\boldsymbol{\psi}$ may include various models for generating diverse data types, including randomly initialized models and pre-trained models obtained from the internet (referred to as "prior models").

Interestingly, it has been observed that using a randomly initialized model to generate initial targets in (2) is more effective and facilitates faster training convergence compared to using random noise (see our analysis in Appendix D). Hence, in this paper, when dealing with unlabeled datasets without specific instructions, DELA opts for a randomly initialized model as the data-initializer.

(b) Updating the model for one step. A batch of data $D_i \subset D$ is randomly sampled, and the parameters $\boldsymbol{\theta}$ in the model backbone $\boldsymbol{\phi}_{\boldsymbol{\theta}}$ and predictor head $\boldsymbol{p}_{\boldsymbol{\theta}}$ are updated as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim D_i} \left[\ell(\boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{\phi}_{\boldsymbol{\theta}}(\mathbf{x})), \mathbf{y})\right]\,, \tag{3}$$

where ℓ denotes the Cosine Similarity loss function, and η is the learning rate.

(c) Evolving data for one step. Building upon the insights from Sun et al. (2024), which underscores the necessity of aligning samples D_X with targets D_Y to create an optimal dataset D, we aim to refine targets D_Y to enhance their congruence with samples D_X. The refinement process leverages the updated model:

$$D_i \leftarrow \{(\mathbf{x}', \mathbf{y}') \mid \mathbf{x}' = \boldsymbol{T}(\mathbf{x}),\ \mathbf{y}' = \lambda \mathbf{y} + (1-\lambda)\boldsymbol{\phi}_{\boldsymbol{\theta}}(\mathbf{x}),\ (\mathbf{x}, \mathbf{y}) \in D_i\}\,, \tag{4}$$

where λ is a blending parameter (refer to Section 4.6 for detailed settings), and $\boldsymbol{T}$ denotes data augmentation technique. Importantly, as $\boldsymbol{\phi}_{\boldsymbol{\theta}}(\mathbf{x})$ is computed during the model update in (3), we utilize these results directly, obviating the need for recomputation in (4). This algorithm is so simple that it can be implemented with two lines of PyTorch code (see Appendix B).

Analysis. Theorem 1 analyzes the convergence of DELA. See proof in Appendix A .

Theorem 1 (Informal statement of convergence of DELA) . *By utilizing a mixture of Gaussian distributions for the initial dataset D_0 and initializing a linear model f_{θ_0}, we demonstrate, under a mild assumption, that $\theta_t \to \theta_\star$ and $D_t \to D_\star$ as $t \to \infty$, where $\star$ denotes the optimal state.*

4 EXPERIMENTS

This section outlines the experimental setup and procedures used to test our hypotheses and evaluate the effectiveness of our proposed methodologies.

4.1 EXPERIMENTAL SETTING

Outlined below are the experimental settings. For further details, refer to Appendix C .

Datasets. For low-resolution data, specifically 32×32, we evaluate our method using the CIFAR-10 and CIFAR-100 datasets (Krizhevsky et al., 2009b;a). To assess scalability and effectiveness on more complex and varied datasets, we also conduct experiments on high-resolution data. This includes the Tiny-ImageNet dataset with 64×64 resolution (Le & Yang, 2015) and the full ImageNet-1K dataset with 224×224 resolution (Deng et al., 2009). Additional details are provided in Appendix C .

Neural network architectures. In alignment with previous studies on dataset distillation (Sun et al., 2024) and self-supervised learning (Susmelj et al., 2020; Da Costa et al., 2022), we employ a variety of backbone architectures to evaluate the generalizability of our method. These include ResNet-{18, 50} (He et al., 2016) and ViT (Dosovitskiy et al., 2020). This selection encompasses a spectrum of model complexities and capacities, facilitating a thorough assessment of our approach.

Baselines. Referring to a widely-used benchmark (Susmelj et al., 2020; Da Costa et al., 2022; Wang et al., 2024b), we evaluate the effectiveness and versatility of our algorithm, DELA, in addressing various data types. We perform comparisons against state-of-the-art methods in three key areas:

- *Self-supervised learning:* We consider methods that enable training models on unlabeled data, including Barlow Twins (Zbontar et al., 2021), BYOL (Jean-Bastien et al., 2020), DINO (Caron et al., 2021), SimSiam (Chen & He, 2021), MoCo (He et al., 2020), SimCLR (Chen et al., 2020a), DCL (Yeh et al., 2022), and NNCLR (Dwibedi et al., 2021).
- *Noisy label learning:* We benchmark against techniques designed to handle noisy labels, including CDR (Xia et al., 2020), SIGUA (Han et al., 2020a), and TNLPAD (Wang et al., 2024b).

Evaluation. In the primary experiments, adhering to established benchmarks and prior research (Susmelj et al., 2020; Da Costa et al., 2022; Chen et al., 2020a; Bardes et al., 2021), we evaluate all trained models using both offline and online linear probing strategies. This approach assesses the representational capacity of the models and ensures a fair and comprehensive comparison with baseline methods. For comparisons with noisy label learning methods, we adopt the training strategy outlined in previous studies (Wang et al., 2024b; Xia et al., 2020).

Table 1: **Evaluation of evolved datasets generated by DELA**. We assess the performance of DELA by training and evaluating models on both the original unlabeled datasets and the evolved datasets. This evaluation is conducted across three model architectures, i.e., ResNet-18, ResNet-50 and ViT-T/16. The datasets include four types: the original dataset and three evolved datasets, denoted as RN18-E, RN50-E, and VT16-E, which correspond to the data evolved using DELA across the respective model architectures. Instances marked with '#' indicate the use of a 50% training budget or training steps. The evaluations are performed across four datasets, i.e., CIFAR-10 (CF-10), CIFAR-100 (CF-100), Tiny-ImageNet (T-IN), and ImageNet-1K (IN-1K).

			DELA training over evolved data:					
Dataset	Architecture	Original	RN18-E	RN50-E	VT16-E	RN18-E#	RN50-E#	VT16-E#
CF-10	ResNet-18	85.3 ± 0.0	88.2 ± 0.0	**88.3 ± 0.0**	86.5 ± 0.0	86.2 ± 0.0	**86.2 ± 0.0**	84.8 ± 0.0
	ResNet-50	88.2 ± 0.0	**90.9 ± 0.1**	**90.9 ± 0.0**	90.0 ± 0.0	89.0 ± 0.0	**89.3 ± 0.1**	87.6 ± 0.0
	ViT-T/16	77.3 ± 0.1	**81.2 ± 0.1**	80.9 ± 0.1	80.7 ± 0.1	**78.0 ± 0.1**	77.4 ± 0.0	77.0 ± 0.0
CF-100	ResNet-18	60.7 ± 0.0	**64.1 ± 0.1**	64.0 ± 0.1	62.7 ± 0.1	**61.8 ± 0.2**	61.2 ± 0.1	60.5 ± 0.1
	ResNet-50	65.0 ± 0.2	**68.9 ± 0.1**	68.7 ± 0.1	68.4 ± 0.1	**66.5 ± 0.0**	66.5 ± 0.1	65.8 ± 0.1
	ViT-T/16	49.4 ± 0.1	54.1 ± 0.1	**54.9 ± 0.0**	54.6 ± 0.1	49.5 ± 0.1	50.4 ± 0.1	**50.8 ± 0.0**
T-IN	ResNet-18	45.7 ± 0.1	48.1 ± 0.1	**48.5 ± 0.0**	48.4 ± 0.1	47.4 ± 0.1	**47.8 ± 0.1**	46.9 ± 0.0
	ResNet-50	51.9 ± 0.1	**55.2 ± 0.1**	55.0 ± 0.0	54.6 ± 0.0	53.3 ± 0.1	**53.6 ± 0.1**	52.6 ± 0.0
	ViT-T/16	39.5 ± 0.1	**42.7 ± 0.1**	**42.7 ± 0.1**	42.5 ± 0.1	39.9 ± 0.1	**40.1 ± 0.1**	39.5 ± 0.1
IN-1K	ResNet-50	59.9 ± 0.0	-	**63.4 ± 0.0**	62.6 ± 0.1	-	**61.9 ± 0.0**	60.4 ± 0.0
	ViT-T/16	46.6 ± 0.0	-	**53.8 ± 0.0**	52.4 ± 0.0	-	**53.4 ± 0.0**	51.1 ± 0.0

Implementation details. We implement our method by extending a popular self-supervised learning open-source benchmark (Susmelj et al., 2020) and a noisy label learning approach (Wang et al., 2024b). We utilize a fair configuration for all our experiments. This includes using AdamW as the optimizer with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e-8$, weight decay of 0.01, and a mini-batch size of 128 (except for ImageNet-1K, where we use a mini-batch size of 512). Our implementation is conducted using PyTorch (Paszke et al., 2019), and all experiments are performed on NVIDIA RTX 4090 and H100 GPUs. See more detailed configurations and hyper-parameters in Appendix C.

4.2 Efficient and Effective Model Training Using Evolved Data from DELA

Recall that our DELA algorithm can enhance the data quality during the learning process, resulting in data that is more refined and beneficial for subsequent training with different model architectures. To validate the effectiveness and efficiency of the evolved data from DELA, we first apply DELA learning on original unlabeled datasets using specific model architectures. Subsequently, we evaluate the evolved datasets across a range of other model architectures. The experimental results presented in Table 1 demonstrate the following:

(a) Given the same training budget, models trained on evolved datasets exhibit superior performance compared to those trained on the original datasets;
(b) Models trained on evolved datasets achieve comparable performance to those trained on original datasets, but with a reduced training budget;
(c) Evolving a dataset with a simpler architecture, such as ResNet-18, can enhance learning effectiveness and efficiency for more complex models like ResNet-50 and ViT-T/16.
(d) Datasets evolved using a ResNet-based architecture, such as RN18-E, typically yield greater benefits and efficiency enhancements for downstream training compared to those evolved with VT16-E. It is reasonable to conjecture that DELA-trained models achieve higher performance with ResNet-based architectures, suggesting that the evolved data is of higher quality.

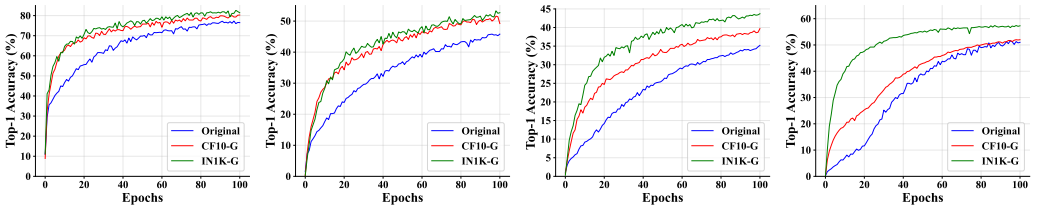
Summary. *Datasets evolved using our* DELA *demonstrate improved efficiency and effectiveness compared to the original datasets. These enhancements are influenced by the evolving process.*

4.3 Accelerating DELA by Using Non-Randomly Initialized Targets

Recall that our DELA, as illustrated in Section 3, can leverage any prior model to initialize targets for given unlabeled datasets, thereby accelerating the training process. Table 2 showcases the effectiveness and efficiency of our approach in facilitating the learning of robust representations by effectively leveraging the prior knowledge within the prior models. Overall, DELA *consistently outperforms the original* DELA when trained on partial data. For instance, on the Tiny-ImageNet

Table 2: **Accelerating our DELA using various prior models**. We compare evaluation results of the models trained using (a) DELA with 20%, 40% and 60% data; (b) DELA with different prior models; (c) DELA with full data, denoted as DELA* in this table. Regarding the prior models used for our DELA, we respectively utilize six models with increasing representation capabilities, including (a) four DELA*-trained models (CF10-T, CF100-T, TIN-T, IN1K-T) corresponding to four datasets (listed below); (b) CLIP-RN50; and (c) YOLOv9. We underline the results that outperform the full data training, and **bold** the results that achieve the highest performance using a specific ratio of data. All the networks used for training are ResNet-18, except the ResNet-50 used for IN-1K.

			DELA initializes targets w/						
Dataset	%	DELA	CF10-T	CF100-T	TIN-T	IN1K-T	CLIP-RN50	YOLOv9	DELA*
CF-10	20	76.8 ± 0.0	79.7 ± 0.1	79.1 ± 0.0	78.9 ± 0.0	**81.5 ± 0.1**	78.1 ± 0.0	77.9 ± 0.0	85.7 ± 0.1
	40	80.6 ± 0.0	83.1 ± 0.0	83.1 ± 0.1	82.3 ± 0.0	**84.8 ± 0.1**	82.5 ± 0.1	81.7 ± 0.0	
	60	83.2 ± 0.1	85.2 ± 0.1	84.8 ± 0.0	84.2 ± 0.0	**86.6 ± 0.1**	84.6 ± 0.0	83.8 ± 0.1	
CF-100	20	49.7 ± 0.1	54.0 ± 0.2	53.7 ± 0.1	53.2 ± 0.1	**54.6 ± 0.1**	52.0 ± 0.1	50.5 ± 0.1	60.4 ± 0.1
	40	54.8 ± 0.0	58.5 ± 0.0	59.0 ± 0.1	58.6 ± 0.1	**59.9 ± 0.1**	57.6 ± 0.2	56.5 ± 0.1	
	60	57.4 ± 0.1	61.4 ± 0.1	61.2 ± 0.1	60.4 ± 0.1	**62.0 ± 0.2**	60.5 ± 0.0	59.1 ± 0.2	
T-IN	20	39.0 ± 0.0	42.0 ± 0.0	42.3 ± 0.1	42.8 ± 0.0	**45.2 ± 0.2**	42.0 ± 0.1	40.6 ± 0.1	44.9 ± 0.2
	40	42.2 ± 0.1	46.0 ± 0.1	46.3 ± 0.1	46.1 ± 0.0	**48.3 ± 0.0**	46.4 ± 0.1	43.9 ± 0.1	
	60	44.2 ± 0.1	47.0 ± 0.1	46.8 ± 0.0	46.9 ± 0.0	**50.1 ± 0.1**	47.9 ± 0.1	46.0 ± 0.0	
IN-1K	20	53.9 ± 0.1	54.7 ± 0.0	54.8 ± 0.1	55.3 ± 0.1	58.8 ± 0.0	**62.6 ± 0.0**	56.1 ± 0.0	59.9 ± 0.0
	40	56.9 ± 0.0	57.3 ± 0.1	57.2 ± 0.0	57.4 ± 0.0	61.9 ± 0.0	**65.0 ± 0.0**	59.7 ± 0.0	
	60	58.1 ± 0.0	57.7 ± 0.0	57.8 ± 0.1	58.1 ± 0.1	62.1 ± 0.1	**66.2 ± 0.0**	60.0 ± 0.1	



(a) CIFAR-10 (b) CIFAR-100 (c) Tiny-ImageNet (d) ImageNet-1K

Figure 3: **Convergence analysis of DELA with different prior models across datasets.** This figure illustrates the accuracy convergence of DELA when initialized with three distinct methods for generating targets: Original refers to our default strategy of using a randomly initialized model to generate initial targets. CF10-G represents targets generated by the CIFAR-10 pre-trained model, while IN1K-G denotes targets generated by an ImageNet-1K pre-trained model. Each subplot corresponds to a specific evaluation dataset. All experiments were conducted using 40% of the full training data to assess performance under limited data scenarios.

dataset, DELA, when using a model pre-trained on CIFAR-10 as the prior model and leveraging only 40% of the data, can outperform DELA-trained models that utilize the entire dataset. Specifically:

(a) Various robust prior models, such as CLIP-RN50 and IN1K-T, even when trained on task-agnostic datasets like YOLOv9[1], significantly enhance the training efficiency and performance of DELA compared to its original version;
(b) In specific scenarios, such as training on Tiny-ImageNet, DELA demonstrates resilience to prior knowledge choice. Employing CF10-T as the prior model yields performance competitive with models trained on extensive datasets like CLIP-RN50.

Summary. *Our* DELA *effectively exploits prior knowledge from various pre-trained models, whether task- or architecture-agnostic, downloaded from the internet or personal repositories. This capability enhances data initialization, thus accelerating the training process.*

Learning curve analysis. To further elucidate the impact of non-randomly initialized targets, we conducted a comparative analysis of convergence between DELA using different prior models.

Figure 3 illustrates the convergence dynamics of DELA when initialized with different methods across four datasets. Our analysis reveals several key insights:

[1]We employ the backbone of YOLOv9 as the prior model due to its superior capability for representation extraction (Wang et al., 2024a).

Table 3: **Evaluating our DELA against noisy-label learning methods.** We evaluate the performance of our algorithm, DELA, by training and assessing models on noise-labeled datasets with different noise levels (denoted by % in this table). This evaluation is conducted against four established methods. The assessment encompasses four datasets: MNIST (MT), Fashion-MNIST (F-MT), CIFAR-10 (CF-10) and CIFAR-100 (CF-100).

Dataset	%	ResNet-18					ResNet-50				
		CDR	SIGUA	TNLPAD	Standard	DELA	CDR	SIGUA	TNLPAD	Standard	DELA
MT	20	98.2 ± 0.0	98.7 ± 0.0	98.7 ± 0.0	98.5 ± 0.0	**99.3 ± 0.0**	98.0 ± 0.0	97.0 ± 0.0	98.5 ± 0.1	**99.7 ± 0.0**	99.4 ± 0.0
	40	97.8 ± 0.0	98.1 ± 0.0	93.7 ± 0.0	95.2 ± 0.1	**99.2 ± 0.0**	98.9 ± 0.0	98.0 ± 0.0	97.0 ± 0.0	98.9 ± 0.1	**99.4 ± 0.0**
	60	91.1 ± 0.0	97.4 ± 0.0	83.4 ± 0.2	84.3 ± 0.1	**99.3 ± 0.0**	97.5 ± 0.0	92.8 ± 0.0	85.8 ± 0.2	96.4 ± 0.0	**99.4 ± 0.0**
F-MT	20	**94.0 ± 0.1**	90.8 ± 0.0	90.2 ± 0.0	90.1 ± 0.0	92.1 ± 0.0	93.3 ± 0.1	89.7 ± 0.0	90.3 ± 0.1	91.5 ± 0.0	**93.6 ± 0.0**
	40	87.9 ± 0.2	88.8 ± 0.0	84.5 ± 0.1	85.0 ± 0.0	**91.9 ± 0.0**	86.7 ± 0.0	87.3 ± 0.0	85.6 ± 0.0	88.6 ± 0.0	**93.4 ± 0.0**
	60	83.8 ± 0.1	83.2 ± 0.0	75.3 ± 0.2	77.9 ± 0.1	**91.2 ± 0.0**	83.0 ± 0.2	76.3 ± 0.0	81.9 ± 0.1	86.3 ± 0.2	**93.4 ± 0.1**
CF-10	20	89.1 ± 0.0	74.6 ± 0.1	**89.5 ± 0.0**	88.1 ± 0.0	85.3 ± 0.1	**89.9 ± 0.1**	64.5 ± 0.1	**89.9 ± 0.0**	89.5 ± 0.1	87.5 ± 0.0
	40	**84.7 ± 0.1**	60.2 ± 0.1	84.4 ± 0.1	82.9 ± 0.1	84.5 ± 0.0	85.6 ± 0.1	37.9 ± 0.0	**86.0 ± 0.1**	84.7 ± 0.1	86.0 ± 0.0
	60	78.1 ± 0.1	23.4 ± 0.1	68.2 ± 0.2	74.2 ± 0.1	**82.7 ± 0.1**	78.2 ± 0.2	20.5 ± 0.6	69.1 ± 0.1	78.8 ± 0.1	**83.3 ± 0.0**
CF-100	20	**66.0 ± 0.1**	30.2 ± 0.1	61.6 ± 0.0	61.3 ± 0.1	60.3 ± 0.1	66.6 ± 0.2	20.5 ± 0.1	63.4 ± 0.1	**67.3 ± 0.0**	65.2 ± 0.1
	40	**60.2 ± 0.1**	16.0 ± 0.0	57.4 ± 0.1	53.6 ± 0.2	58.3 ± 0.1	57.8 ± 0.2	8.6 ± 0.1	55.7 ± 0.0	61.0 ± 0.0	**62.8 ± 0.1**
	60	42.0 ± 0.2	4.1 ± 0.0	42.1 ± 0.0	44.7 ± 0.1	**56.6 ± 0.1**	47.2 ± 0.1	3.8 ± 0.1	46.2 ± 0.2	53.0 ± 0.1	**59.2 ± 0.1**

(a) DELA, initialized with non-random prior models, consistently converges faster across all datasets, achieving higher accuracy in fewer epochs compared to random initialization.
(b) DELA with strong priors consistently outperforms those with weaker priors. Notably, on complex datasets such as Tiny ImageNet and ImageNet-1K, the performance gap is more pronounced.

4.4 LEARNING AND DENOISING DATASETS WITH NOISY TARGETS USING DELA

To further investigate the potential applications of DELA algorithm in addressing challenges in contemporary deep learning, we apply DELA to learning tasks involving noise-labeled datasets. Following the strategy outlined in Wang et al. (2024b), we introduce symmetric noise at varying levels to the clean targets within the original datasets and benchmark various methods, including DELA. The experimental results in Table 3 and Appendix E indicate that:

(a) DELA consistently outperforms several SOTA baselines across multiple datasets;
(b) DELA demonstrates robustness to varying degrees of noise. For instance, while the performance of TNLPAD significantly declines with increased noise levels (from 20% to 40%), DELA exhibits only minor performance degradation.

Our DELA has exhibited twofold advantages: (a) It can adapt to and evolve various types of original targets (see Section 3), including learning directly from noisy datasets; (b) The evolved noisy datasets, considered as cleaner versions, can be effectively utilized to train models using DELA.

4.5 DELA BENCHMARKS AGAINST VARIOUS SELF-SUPERVISED LEARNING METHODS

As we discussed in Section 3 , our DELA can deal with unlabeled data by initializing targets using random models. Therefore, to demonstrate the effectiveness and versatility of DELA in self-supervised learning against various baselines, we conduct experiments with widely-used techniques in Table 4 . The results reveal that DELA consistently achieves superior or comparable performance to SOTA methods across diverse datasets and neural network architectures, underscoring its robust generalization capability.

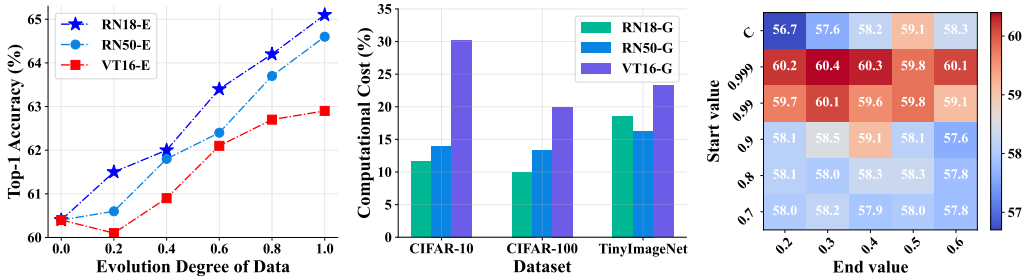
4.6 ABLATION STUDY

In this section, we present a comprehensive ablation study to analyze the impact of various components and design choices on the performance of DELA. We focus on three key aspects: the influence of evolved data, the computational efficiency gained through incorporating prior knowledge, and the sensitivity of the algorithm to different λ scheduler configurations.

Impact of evolved data on DELA. The evolved data in DELA serves as a powerful mechanism for incorporating various priors to accelerate training. As the strength of these priors gradually increases, we observe a corresponding enhancement in the final performance. Figure 4a illustrates this trend, showcasing the performance of DELA using different evolved data sources on the CIFAR-100 dataset.

Table 4: **Evaluating our DELA against self-supervised learning methods.** We analyze our DELA by training and evaluating models over unlabeled datasets, against seven conventional self-supervised learning methods.

Dataset	Architecture	Barlow	BYOL	DINO	MoCo	SimCLR	DCL	NNCLR	DELA
CF-10	ResNet-18	84.8 ± 0.0	84.1 ± 0.0	81.2 ± 0.0	83.5 ± 0.1	83.6 ± 0.0	84.4 ± 0.0	82.4 ± 0.1	**85.7 ± 0.1**
	ResNet-50	86.4 ± 0.1	85.4 ± 0.1	83.9 ± 0.1	85.5 ± 0.1	86.2 ± 0.0	87.4 ± 0.0	82.5 ± 0.1	**87.5 ± 0.0**
	ViT-T/16	72.6 ± 0.1	72.2 ± 0.0	76.7 ± 0.0	75.6 ± 0.0	75.7 ± 0.0	**78.9 ± 0.1**	69.5 ± 0.1	77.1 ± 0.0
CF-100	ResNet-18	58.4 ± 0.1	57.4 ± 0.1	51.0 ± 0.0	58.6 ± 0.1	55.5 ± 0.0	59.9 ± 0.1	46.4 ± 0.1	**60.4 ± 0.1**
	ResNet-50	62.7 ± 0.1	59.1 ± 0.1	57.1 ± 0.1	63.0 ± 0.0	60.8 ± 0.0	63.6 ± 0.1	48.2 ± 0.1	**63.8 ± 0.3**
	ViT-T/16	42.7 ± 0.0	40.7 ± 0.1	44.3 ± 0.1	48.1 ± 0.1	45.0 ± 0.1	**51.8 ± 0.1**	32.7 ± 0.0	48.3 ± 0.2
T-IN	ResNet-18	44.2 ± 0.0	44.3 ± 0.1	36.1 ± 0.0	42.4 ± 0.2	41.5 ± 0.1	44.6 ± 0.0	36.4 ± 0.1	**44.9 ± 0.2**
	ResNet-50	**51.7 ± 0.1**	47.7 ± 0.2	42.9 ± 0.1	49.7 ± 0.2	47.8 ± 0.1	49.9 ± 0.3	41.9 ± 0.1	49.9 ± 0.0
	ViT-T/16	**39.7 ± 0.1**	29.6 ± 0.1	30.6 ± 0.0	37.3 ± 0.0	31.6 ± 0.0	37.9 ± 0.2	20.3 ± 0.1	37.9 ± 0.1
IN-1K	ResNet-50	59.6 ± 0.0	**62.8 ± 0.0**	52.2 ± 0.0	57.6 ± 0.0	58.0 ± 0.0	60.6 ± 0.0	58.3 ± 0.1	59.9 ± 0.0
	ViT-T/16	43.1 ± 0.0	43.9 ± 0.0	40.4 ± 0.0	43.5 ± 0.0	40.8 ± 0.1	**46.6 ± 0.0**	35.5 ± 0.1	**46.6 ± 0.0**



(a) Data evolution. (b) Computational cost. (c) λ scheduler.

Figure 4: **Comprehensive ablation study of DELA.** (a) Performance of DELA using different evolved data sources on CIFAR-100. RN18-E, RN50-E, and VT16-E represent evolved data obtained by training ResNet18, ResNet50, and ViT-T/16 architectures, respectively, on the CIFAR-100 dataset. (b) Analysis of computational efficiency using different prior models. The y-axis indicates the percentage of training budget/steps required to achieve the same accuracy as the original DELA. RN18-G, RN50-G, and VT16-G represent targets generated by ResNet18, ResNet50, and ViT-T/16 models, respectively. (c) Sensitivity analysis of λ scheduler configurations on CIFAR-100. The heatmap shows the final accuracy for different combinations of starting and ending λ values. "C" indicates a constant λ throughout training: for example, a cell with y-axis "C" and x-axis 0.4 means λ remains constant at 0.4 for the entire training process.

In our experiments, we define the evolution degree as the extent of training progression in the co-evolving process. Specifically, with our standard setting of 100 epochs for full evolution, an evolution degree of 0.1 corresponds to the dataset saved after 10 epochs of training. This metric allows us to quantify the impact of evolutionary progression on model performance. Our analysis reveals:

(a) The performance of DELA improves substantially as the evolution degree increases, regardless of the architecture used to generate the evolved data. This demonstrates the robustness of our method across different data sources and suggests that the co-evolution process consistently enhances the quality of the training data.
(b) The performance trajectories for different architectures show similar upward trends, further emphasizing the consistency of our method's improvement across various data sources.
(c) It is interesting to note that the ViT-based evolved data (VT16-E) demonstrates the least improvement, which might be attributed to the convolutional inductive bias present in ResNet architectures. Nevertheless, it still exhibits a noteworthy 2.5 percentage point improvement at best.

Summary. *The performance of* DELA *consistently improves with higher evolution degrees across data evolved from various neural network architectures, demonstrating its superior robustness.*

Analysis of computational cost in DELA by leveraging different prior models. Leveraging co-evolved data during training allows us to substantially reduce training time, thereby conserving computational resources. Figure 4b illustrates the efficiency gains across different datasets and prior knowledge sources. Our analysis reveals:

(a) In all datasets, the incorporation of prior knowledge significantly reduces the training budget/steps needed to achieve the same accuracy as a model trained with the original DELA.
(b) The selection of prior knowledge sources affects efficiency variably across datasets. Specifically, RN50-G demonstrates optimal efficiency for CIFAR-100 and TinyImageNet, whereas RN18-G is most effective for CIFAR-10.

Summary. *Our* DELA*'s capability to efficiently integrate various prior models—many of which are freely available online—offers a significant advantage in training efficiency and resource utilization. These findings underscore the potential of our approach to substantially reduce computational costs, especially with complex datasets where traditional model-centric methods demand extensive resources. This highlights the promise of our proposed data-centric learning paradigm (see Definition 1) and corresponding framework* DELA *for the community.*

Sensitivity analysis of λ scheduler in DELA. The λ scheduler in DELA plays a crucial role in balancing the impact of prior model knowledge on the training process. It controls how the incorporated prior, whether from co-evolved data, randomly initialized models, or pre-trained models, affects the evolution process. Larger λ values allow for more effective utilization of the prior, especially in the early stages of training, but may lead to over-fitting if maintained throughout the process.

In our analysis, we considered two primary λ scheduler configurations: constant settings, where λ remains unchanged throughout training, and cosine annealing schedules, where λ decreases gradually from a higher initial value to a lower final value. This approach allows us to examine the impact of both static and dynamic prior influence on the training process.

Figure 4c and Figure 5 illustrate the performance of DELA under various λ scheduler configurations on the CIFAR-100 dataset. Figure 4c represents the scenario using initial targets generated by a randomly initialized model (weak prior), while Figure 5 shows results using evolved data (RN18-E), which are derived from a ResNet18 model trained on CIFAR-100 (strong prior). Our analysis reveals:

(a) For both weak and strong priors, higher starting λ values (0.99 and 0.999) generally lead to better performance, indicating the importance of leveraging prior model knowledge.
(b) The impact of the ending λ value is less pronounced, particularly for strong priors, suggesting that the method is robust to this parameter within the tested range (0.2 to 0.6).
(c) Constant λ values (denoted by "C" in the first row of the heatmaps) consistently underperform compared to annealing schedules. This observation highlights the benefits of dynamically adjusting the influence of prior knowledge throughout the training process, allowing the model to initially leverage the prior heavily and then gradually adapt to the specific dataset.
(d) The performance gap between weak and strong priors is substantial (approximately 5 percentage points), emphasizing the value of incorporating high-quality prior knowledge.

Summary. *We recommend that practitioners use a high starting λ value (0.999) with a gradual decrease to a moderate ending value (around 0.3) when employing the cosine annealing schedule. This configuration allows the training of* DELA *to benefit significantly from prior knowledge in the early stages while still adapting effectively to the specific dataset as training progresses.*

Furthermore, please refer to Appendix D for additional ablation studies of our DELA.

5 LIMITATION AND CONCLUSION

This paper introduces data-evolution learning, a novel data-centric learning paradigm that simultaneously evolves datasets and models. Our proposed DELA demonstrates significant advantages over traditional model-centric approaches by generating reusable optimized datasets, exhibiting versatile compatibility with various data types, and effectively leveraging prior knowledge to accelerate learning. Extensive experiments across multiple datasets, architectures, and baselines validate DELA's effectiveness and efficiency, consistently matching or outperforming SOTA model-centric methods in various learning scenarios. The data-evolution learning paradigm opens new possibilities for representation learning, providing a fresh perspective on addressing data quality and computational efficiency challenges. While results are promising, future work could explore theoretical foundations, extend to more diverse tasks, and investigate potential in other domains such as natural language processing and reinforcement learning.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. Sequential modeling enables scalable learning for large vision models. *arXiv preprint arXiv:2312.00785*, 2023.

Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning. *arXiv preprint arXiv:2105.04906*, 2021.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Yue Cao, Zhenda Xie, Bin Liu, Yutong Lin, Zheng Zhang, and Han Hu. Parametric instance classification for unsupervised visual feature learning. In *NeurIPS*, 2020.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In *NeurIPS*, 2020.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4750–4759, 2022.

Hao Chen, Jindong Wang, Lei Feng, Xiang Li, Yidong Wang, Xing Xie, Masashi Sugiyama, Rita Singh, and Bhiksha Raj. A general framework for learning from weak supervision. *arXiv preprint arXiv:2402.01922*, 2024.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020a.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020b.

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. *arXiv preprint arXiv:2104.02057*, 2021.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.

Justin Cui, Ruochen Wang, Si Si, and Cho-Jui Hsieh. Dc-bench: Dataset condensation benchmark. *Advances in Neural Information Processing Systems*, 35:810–822, 2022.

Justin Cui, Ruochen Wang, Si Si, and Cho-Jui Hsieh. Scaling up dataset distillation to imagenet-1k with constant memory. In *International Conference on Machine Learning*, pp. 6565–6590. PMLR, 2023.

Victor Guilherme Turrisi Da Costa, Enrico Fini, Moin Nabi, Nicu Sebe, and Elisa Ricci. solo-learn: A library of self-supervised methods for visual representation learning. *Journal of Machine Learning Research*, 23(56):1–6, 2022.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Jiawei Du, Yidi Jiang, Vincent YF Tan, Joey Tianyi Zhou, and Haizhou Li. Minimizing the accumulated trajectory error to improve dataset distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3749–3758, 2023.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9588–9597, 2021.

Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111:98–136, 2015.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2018.

Samir Yitzhak Gadre, Gabriel Ilharco, Alex Fang, Jonathan Hayase, Georgios Smyrnis, Thao Nguyen, Ryan Marten, Mitchell Wortsman, Dhruba Ghosh, Jieyu Zhang, et al. Datacomp: In search of the next generation of multimodal datasets. *arXiv preprint arXiv:2304.14108*, 2023.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *International Conference on Learning Representations*, 2017.

Ziyao Guo, Kai Wang, George Cazenavette, Hui Li, Kaipeng Zhang, and Yang You. Towards lossless dataset distillation via difficulty-aligned trajectory matching. *arXiv preprint arXiv:2310.05773*, 2023.

Bo Han, Gang Niu, Xingrui Yu, Quanming Yao, Miao Xu, Ivor Tsang, and Masashi Sugiyama. Sigua: Forgetting may make learning with noisy labels more robust. In *International Conference on Machine Learning*, pp. 4006–4016. PMLR, 2020a.

Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*, 2020b.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

Qianjiang Hu, Xiao Wang, Wei Hu, and Guo-Jun Qi. Adco: Adversarial contrast for efficient learning of unsupervised representations from self-trained negative adversaries. In *CVPR*, 2021.

Grill Jean-Bastien, Strub Florian, Altché Florent, Tallec Corentin, Pierre Richemond H., Buchatskaya Elena, Doersch Carl, Bernardo Pires Avila, Zhaohan Guo Daniel, Mohammad Azar Gheshlaghi, Piot Bilal, Kavukcuoglu Koray, Munos Rémi, and Valko Michal. Bootstrap your own latent - a new approach to self-supervised learning. In *NeurIPS*, 2020.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pp. 2304–2313, 2018.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *NeurIPS*, 2020.

Jang-Hyun Kim, Jinuk Kim, Seong Joon Oh, Sangdoo Yun, Hwanjun Song, Joonhyun Jeong, Jung-Woo Ha, and Hyun Oh Song. Dataset condensation via efficient synthetic-data parameterization. In *International Conference on Machine Learning*, pp. 11102–11118. PMLR, 2022.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009a.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 and cifar-100 datasets. *URl: https://www. cs. toronto. edu/kriz/cifar. html*, 6(1):1, 2009b.

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5447–5456, 2018.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. In *ICLR*, 2021.

Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *IEEE International Conference on Computer Vision*, pp. 1910–1918, 2017.

Tongliang Liu and Dacheng Tao. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461, 2015.

Yanqing Liu, Jianyang Gu, Kai Wang, Zheng Zhu, Wei Jiang, and Yang You. Dream: Efficient dataset distillation by representative matching. *arXiv preprint arXiv:2302.14416*, 2023.

Xingjun Ma, Yisen Wang, Michael E Houle, Shuo Zhou, Sarah Erfani, Shutao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. In *International Conference on Machine Learning*, pp. 3355–3364, 2018.

Fatemehsadat Mireshghallah, Mohammadkazem Taram, Praneeth Vepakomma, Abhishek Singh, Ramesh Raskar, and Hadi Esmaeilzadeh. Privacy in deep learning: A survey. *arXiv preprint arXiv:2004.12254*, 2020.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Tiago P Pagano, Rafael B Loureiro, Fernanda VN Lisboa, Rodrigo M Peixoto, Guilherme AS Guimarães, Gustavo OR Cruz, Maira M Araujo, Lucas L Santos, Marco AS Cruz, Ewerton LS Oliveira, et al. Bias and unfairness in machine learning models: a systematic review on datasets, tools, fairness metrics, and identification and mitigation methods. *Big data and cognitive computing*, 7(1):15, 2023.

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.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11893–11902, 2020.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pp. 4334–4343, 2018.

Pierre H Richemond, Jean-Bastien Grill, Florent Altché, Corentin Tallec, Florian Strub, Andrew Brock, Samuel Smith, Soham De, Razvan Pascanu, Bilal Piot, et al. Byol works even without batch statistics. *arXiv preprint arXiv:2010.10241*, 2020.

Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. Imagenet-21k pretraining for the masses. *arXiv preprint arXiv:2104.10972*, 2021.

Burr Settles, Mark Craven, and Lewis Friedland. Active learning with real annotation costs. In *Proceedings of the NIPS workshop on cost-sensitive learning*, volume 1. Vancouver, CA:, 2008.

Shitong Shao, Zeyuan Yin, Muxin Zhou, Xindong Zhang, and Zhiqiang Shen. Generalized large-scale data condensation via various backbone and statistical matching. *arXiv preprint arXiv:2311.17950*, 2023.

Avi Singh, John D Co-Reyes, Rishabh Agarwal, Ankesh Anand, Piyush Patil, Peter J Liu, James Harrison, Jaehoon Lee, Kelvin Xu, Aaron Parisi, et al. Beyond human data: Scaling self-training for problem-solving with language models. *arXiv preprint arXiv:2312.06585*, 2023.

Martin Strobel and Reza Shokri. Data privacy and trustworthy machine learning. *IEEE Security & Privacy*, 20(5):44–49, 2022.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*, 2019.

M. Sugiyama, H. Bao, T. Ishida, N. Lu, T. Sakai, and G. Niu. *Machine Learning from Weak Supervision: An Empirical Risk Minimization Approach*. MIT Press, Cambridge, Massachusetts, USA, 2022.

Peng Sun, Bei Shi, Daiwei Yu, and Tao Lin. On the diversity and realism of distilled dataset: An efficient dataset distillation paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Igor Susmelj, Matthias Heller, Philipp Wirth, Jeremy Prescott, and Malte Ebner et al. Lightly. *GitHub. Note: https://github.com/lightly-ai/lightly*, 2020.

Chenxin Tao, Honghui Wang, Xizhou Zhu, Jiahua Dong, Shiji Song, Gao Huang, and Jifeng Dai. Exploring the equivalence of siamese self-supervised learning via a unified gradient framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14431–14440, 2022.

Tatiana Tommasi, Novi Patricia, Barbara Caputo, and Tinne Tuytelaars. A deeper look at dataset bias. *Domain adaptation in computer vision applications*, pp. 37–55, 2017.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 9446–9454, 2018.

Arash Vahdat. Toward robustness against label noise in training deep discriminative neural networks. In *Conference and Workshop on Neural Information Processing Systems*, 2017.

Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616*, 2024a.

Jingyi Wang, Xiaobo Xia, Long Lan, Xinghao Wu, Jun Yu, Wenjing Yang, Bo Han, and Tongliang Liu. Tackling noisy labels with network parameter additive decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024b.

Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. 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.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *ICML*, 2020.

Tongzhou Wang, Jun-Yan Zhu, Antonio Torralba, and Alexei A Efros. Dataset distillation. *arXiv preprint arXiv:1811.10959*, 2018.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *IEEE International Conference on Computer Vision*, pp. 322–330, 2019.

Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *International conference on learning representations*, 2020.

Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. L_dmi: An information-theoretic noise-robust loss function. In *arXiv*, 2019.

Jiangchao Yao, Jiajie Wang, Ivor W Tsang, Ya Zhang, Jun Sun, Chengqi Zhang, and Rui Zhang. Deep learning from noisy image labels with quality embedding. *IEEE Transactions on Image Processing*, 28(4):1909–1922, 2018.

Chun-Hsiao Yeh, Cheng-Yao Hong, Yen-Chi Hsu, Tyng-Luh Liu, Yubei Chen, and Yann LeCun. Decoupled contrastive learning. In *European conference on computer vision*, pp. 668–684. Springer, 2022.

Zeyuan Yin, Eric Xing, and Zhiqiang Shen. Squeeze, recover and relabel: Dataset condensation at imagenet scale from a new perspective. *arXiv preprint arXiv:2306.13092*, 2023.

Ruonan Yu, Songhua Liu, and Xinchao Wang. Dataset distillation: A comprehensive review. *arXiv preprint arXiv:2301.07014*, 2023.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *ICML*, 2021.

Lei Zhang, Jie Zhang, Bowen Lei, Subhabrata Mukherjee, Xiang Pan, Bo Zhao, Caiwen Ding, Yao Li, and Dongkuan Xu. Accelerating dataset distillation via model augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11950–11959, 2023.

Zhilu Zhang and Mert R Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. In *Conference and Workshop on Neural Information Processing Systems*, 2018.

Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 6514–6523, 2023.

Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. *arXiv preprint arXiv:2006.05929*, 2020.

Ganlong Zhao, Guanbin Li, Yipeng Qin, and Yizhou Yu. Improved distribution matching for dataset condensation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7856–7865, 2023.

Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National science review*, 5(1): 44–53, 2018.

Rui Zhu, Bingchen Zhao, Jingen Liu, Zhenglong Sun, and Chang Wen Chen. Improving contrastive learning by visualizing feature transformation. In *ICCV*, 2021.

A PROOF OF THEOREM 1

We consider a mixture of two Gaussian distributions, denoted as $\mathcal{N}^+(\mu, \sigma)$ and $\mathcal{N}^-(-\mu, \sigma)$, and use a linear model:

$$\phi_\theta(x) = \begin{cases} 1, & \text{if } x < \theta \\ -1, & \text{otherwise} \end{cases}$$

The distribution of samples is defined as follows:

$$G_X = \{(1-y)\cdot x^+ + y \cdot x^- \mid y \sim \text{Bernoulli}(0.5),\, x^+ \sim \mathcal{N}^+,\, x^- \sim \mathcal{N}^-\}$$

(a) Initializing Targets for Given Samples

The initial data G_0 for the model ϕ_{θ_0} is defined by:

$$G_0 = \{(x, y) \mid y = \phi_{\theta_0}(x),\, x \sim G_X\}$$

(b) Updating the Model for One Step

The model ϕ_θ is updated using:

$$\theta_t = \theta_{t-1} - \eta \nabla_\theta \mathbb{E}_{(x,y)\sim G_{t-1},\, \epsilon\sim\mathcal{N}(0,1)} \left[\ell(\phi_\theta(x + \alpha\cdot\epsilon), y)\right]$$

where ℓ is the mean squared error (MSE) loss function, η is the learning rate, t is the training step index, and α determines the degree of data augmentation.

(c) Evolving Data for One Step

The updated model informs the refinement of the data:

$$G_t = \{(x, y) \mid y = \phi_{\theta_t}(x),\, x \sim G_X\}$$

This process is then repeated by reverting to step (b).

We aim to prove that $\theta_t \to 0$ as $t \to \infty$ provided that the initial threshold satisfies $|\mu| > |\theta_0| \approx 0$.

Proof. 1. Smooth Approximation of $\phi_\theta(x)$

The activation function $\phi_\theta(x)$ is non-differentiable at $x = \theta$, which poses challenges for gradient-based optimization. To facilitate differentiation, we approximate $\phi_\theta(x)$ with a smooth surrogate function. A common choice is the hyperbolic tangent function:

$$\phi_\theta^\kappa(x) = \tanh\left(\kappa(\theta - x)\right),$$

where $\kappa > 0$ controls the steepness of the approximation. As $\kappa \to \infty$, $\phi_\theta^\kappa(x)$ approaches the original activation function $\phi_\theta(x)$.

2. Derivation of the Gradient of the Loss Function

We employ the Mean Squared Error (MSE) loss function for a single sample (x, y):

$$\ell(\phi_\theta^\kappa(x + \alpha\epsilon), y) = \frac{1}{2}\left(\phi_\theta^\kappa(x + \alpha\epsilon) - y\right)^2,$$

where α denotes the degree of data augmentation, and $\epsilon \sim \mathcal{N}(0, 1)$ introduces stochastic perturbations.

The expected loss over the dataset G_{t-1} at training step $t-1$ is:

$$\mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim G_{t-1}, \epsilon\sim\mathcal{N}(0,1)}\left[\frac{1}{2}\left(\phi_\theta^\kappa(x + \alpha\epsilon) - y\right)^2\right].$$

Taking the gradient of the expected loss with respect to θ yields:

$$\nabla_\theta \mathcal{L}(\theta) = \mathbb{E}_{(x,y)\sim G_{t-1}, \epsilon\sim\mathcal{N}(0,1)}\left[\left(\phi_\theta^\kappa(x + \alpha\epsilon) - y\right)\cdot \nabla_\theta \phi_\theta^\kappa(x + \alpha\epsilon)\right].$$

Given our smooth approximation:

$$\nabla_\theta \phi_\theta^\kappa(z) = \kappa \cdot \text{sech}^2\left(\kappa(\theta - z)\right),$$

where $z = x + \alpha\epsilon$, and sech^2 is the squared hyperbolic secant function.

3. Exploiting Symmetry in the Data Distribution

The data distribution G_X is a symmetric mixture of two Gaussian distributions centered at μ and $-\mu$, respectively. This symmetry implies that for every sample $x \sim \mathcal{N}^+(\mu, \sigma)$ with label $y = 1$, there exists a corresponding sample $-x \sim \mathcal{N}^-(-\mu, \sigma)$ with label $y = -1$.

This symmetry ensures that certain terms in the gradient expectation will cancel out, simplifying the analysis.

4. Linearizing the Gradient Near $\theta = 0$

Assume that the threshold θ is small relative to μ, i.e., $|\theta| < |\mu|$. Under this assumption, we can perform a first-order Taylor expansion of $\phi_\theta^\kappa(z)$ around $\theta = 0$:

$$\phi_\theta^\kappa(z) \approx \tanh(-\kappa z) + \kappa \cdot \text{sech}^2(-\kappa z) \cdot \theta.$$

Since $\tanh$ is an odd function and sech^2 is even, substituting $z = x + \alpha\epsilon$ and using the symmetry of G_X leads to the cancellation of the zeroth-order terms, leaving:

$$\phi_\theta^\kappa(z) - \phi_\theta^\kappa(x) \approx \kappa \cdot \text{sech}^2(\kappa z) \cdot \theta - \kappa \cdot \text{sech}^2(\kappa x) \cdot \theta.$$

Given the symmetry $x \overset{d}{=} -x$, the expectation simplifies, and higher-order terms in θ can be neglected. Thus, the gradient becomes approximately linear in θ:

$$\nabla_\theta \mathcal{L}(\theta) \approx a \cdot \theta,$$

where

$$a = \kappa^2 \cdot \mathbb{E}_{x,\epsilon}\left[\text{sech}^2(\kappa z) \cdot \text{sech}^2(\kappa(\theta - z))\right] > 0.$$

5. Analyzing the Update Rule to Show Convergence

Using gradient descent, the update rule for θ at each step t is:

$$\theta_t = \theta_{t-1} - \eta \cdot \nabla_\theta \mathcal{L}(\theta_{t-1}) \approx \theta_{t-1} - \eta \cdot a \cdot \theta_{t-1} = (1 - \eta a) \cdot \theta_{t-1}.$$

To ensure convergence, the learning rate η must satisfy:

$$0 < \eta a < 2.$$

Under this condition, the factor $|1 - \eta a| < 1$, which guarantees that $|\theta_t|$ decreases geometrically at each step. Consequently, as $t \to \infty$:

$$\theta_t \to 0.$$

□

B Psuedo-Code for Data-Evolution Learning (DeLA)

The pseudo-code for the Data-Evolution Learning Algorithm (DeLA) is provided in Algorithm 1.

C Experimental Details

C.1 Data Description

We conducted our experiments on four widely-used datasets in computer vision: CIFAR-10, CIFAR-100, TinyImageNet, and ImageNet-1K. Table 5 summarizes the key characteristics of these datasets.

Table 5: Summary of datasets used in our experiments

Dataset	Classes	Training Images	Test Images	Image Size
CIFAR-10	10	50,000	10,000	32x32
CIFAR-100	100	50,000	10,000	32x32
TinyImageNet	200	100,000	10,000	64x64
ImageNet-1K	1,000	1,281,167	50,000	224x224

Algorithm 1 Data-Evolution Learning Algorithm (DeLA)

Require: Dataset $D_X = \{\mathbf{x}_i\}_{i=1}^N \in \mathcal{X}$, Data-initializer $\boldsymbol{\psi} : \mathcal{X} \to \mathbb{R}^k$, Learning rate η, Blending parameter λ

Initialize backbone $\boldsymbol{f_\theta} : \mathcal{X} \to \mathbb{R}^d$, projector $\boldsymbol{h_\theta} : \mathbb{R}^d \to \mathbb{R}^k$, predictor $\boldsymbol{p_\theta} : \mathbb{R}^k \to \mathbb{R}^k$
Initialize dataset: $D \leftarrow \{(\mathbf{x}_i, \boldsymbol{\psi}(\mathbf{x}_i))\}_{i=1}^N$
for mini-batch $D_i = \{(\mathbf{x}_j, \mathbf{y}_j)\}_{j=1}^B \sim D$ **do**
  Compute representations:
  $\mathbf{h} \leftarrow \boldsymbol{h_\theta}(\boldsymbol{f_\theta}(\mathbf{x}))$
  $\mathbf{z} \leftarrow \boldsymbol{p_\theta}(\mathbf{h})$
  Update model:
  $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla_{\boldsymbol{\theta}} \left(\frac{1}{|D_i|} \sum_{j=1}^B \frac{\mathbf{z}_j^\top \mathbf{y}_j}{\|\mathbf{z}_j\| \|\mathbf{y}_j\|} \right)$
  Evolve data:
  $D_i \leftarrow \{(\boldsymbol{T}(\mathbf{x}_j), \mathbf{y}_j + (1-\lambda)\mathbf{h}_j)\}_{j=1}^B$
end for
return Trained backbone $\boldsymbol{f_\theta}$ and evolved dataset D

Table 6: Common training settings across all experiments

Parameter	Value
Number of epochs	100
Learning rate	0.001
Weight decay	0.01
Optimizer	AdamW
Loss function	Negative similarity
Precision	Mixed precision (float16)

C.2 Training and Validation Procedures

We employed consistent training settings across all experiments, as detailed in Table 6. Batch sizes varied depending on the dataset, as shown in Table 7.

For evaluation, we used a linear evaluation protocol. The linear classifier was trained for 100 epochs on the frozen features extracted by the main network, using the batch sizes specified in Table 7.

C.3 Experimental Setup and Environment

All experiments were conducted using PyTorch framework with mixed precision training enabled. We used NVIDIA GPUs for accelerated computations, although the specific hardware details may vary depending on the scale of the experiment.

C.4 Evaluation Metrics and Results Interpretation

Our primary evaluation metric was classification accuracy on the test set after linear evaluation. This metric provides a measure of the quality of the learned representations, as it assesses how well a linear classifier can separate classes using these features.

C.5 Reproducibility

To ensure reproducibility, we have provided detailed hyperparameters and experimental settings in this appendix. Our code, including the implementation of DeLA and the evaluation protocols, will be made available in a public repository upon publication.

D Additional Ablation Studies

Impact of Projector Architecture on DeLA Performance The projector architecture plays a crucial role in DeLA's performance, offering several key benefits: (i) it provides flexibility to control

Table 7: Batch sizes for different datasets

Dataset	Training Batch Size	Linear Eval Batch Size
CIFAR-10	128	128
CIFAR-100	128	128
TinyImageNet	128	128
ImageNet-1K	512	1024

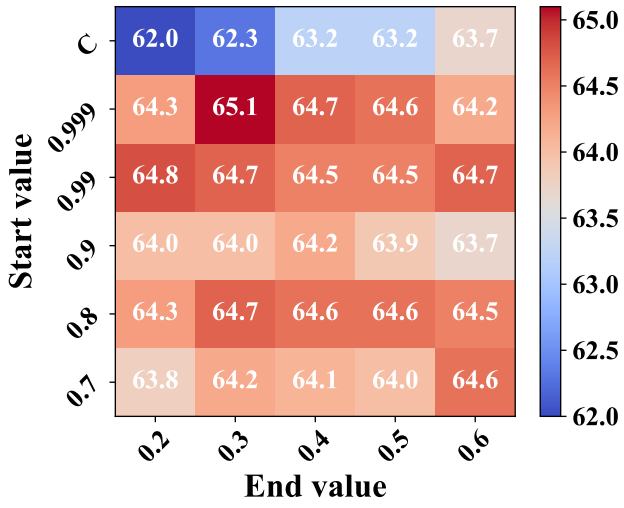


Figure 5: λ **scheduler (strong prior)**. Sensitivity analysis of λ scheduler configurations on CIFAR-100 using strong priors (CF100-RN18-E, evolved data from ResNet18 trained on CIFAR-100). The heatmap shows the final accuracy for different combinations of starting and ending λ values, with "C" indicating constant λ throughout training.

the feature space dimension, (ii) it unifies features from different models, (iii) it strikes a balance between target effectiveness and computational costs, and (iv) it enhances the model's capacity to cluster features effectively. To thoroughly investigate the influence of various projector architectures on DELA's performance, we conducted extensive experiments encompassing several structural variations. Table 8 illustrates the performance of DELA across these architectural variants.

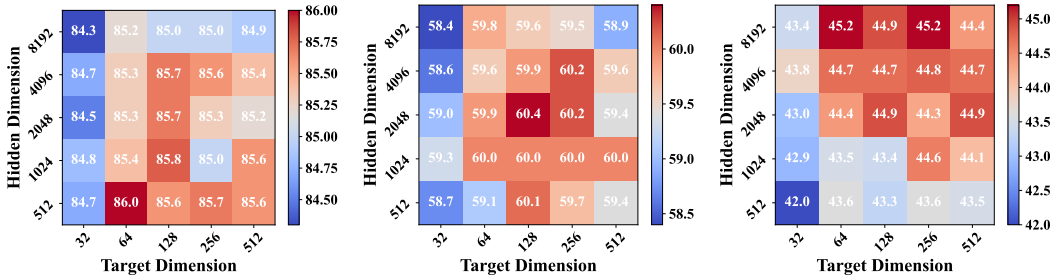
Our analysis reveals several important findings:

(a) The baseline architecture consistently outperforms other variants across all datasets, demonstrating its robustness and effectiveness in diverse learning scenarios. This indicates that each component in the projector architecture is essential and cannot be removed without degrading performance.
(b) Normalization layers, particularly BatchNorm, show consistent benefits across all datasets. This effect is more pronounced in larger, more diverse datasets, suggesting that normalization becomes increasingly crucial as data complexity grows.
(c) Non-linearity, introduced either through activation functions or normalization, is crucial for effective feature projection. This is evidenced by the poor performance of the Linear + Linear architecture across all datasets.
(d) The simplicity of the architecture correlates inversely with its ability to handle complex datasets. Single Linear layer architectures perform adequately on simpler tasks but struggle with more complex datasets, indicating that sophisticated projector architectures are necessary to capture rich feature representations in complex scenarios.

These findings underscore the importance of every parts in DELA's projector, especially when dealing with complex, large-scale datasets. This default configuration consistently outperformed other variants across all datasets, demonstrating its robustness and effectiveness in diverse learning scenarios.

Table 8: **Impact of Projector Architecture on DELA Performance.** We analyze the effect of various projector architectures on the performance of DELA. Results are reported in terms of test accuracy (%) on CIFAR-10, CIFAR-100, and TinyImageNet datasets.

Projector Architecture	CIFAR-10	CIFAR-100	TinyImageNet
Linear + BatchNorm + ReLU + Linear	**85.7**	**60.4**	**44.9**
Linear	82.2	51.4	34.8
Linear + ReLU + Linear	71.7	45.0	<u>39.7</u>
Linear + BatchNorm + Linear	<u>83.4</u>	<u>54.0</u>	36.7
Linear + Linear	10.0	1.0	14.5



(a) CIFAR-10 (b) CIFAR-100 (c) TinyImageNet

Figure 6: **Ablation study on the demensions of projector in DELA.** We illustrate the performance of DELA across various combinations of hidden and output dimensions for the projector.

Furthermore, we examined the sensitivity of DELA to variations in the hidden dimension and output dimension of the projector. Our findings, summarized in Figure 6, reveal:

(a) The impact of projector dimensions varies across datasets, with more complex datasets showing higher sensitivity to dimensional changes. However, the overall performance differences are not drastic, indicating a degree of robustness in the projector architecture.
(b) There is a clear trend of increasing dimensional requirements as dataset complexity increases. Moving from CIFAR-10 to CIFAR-100 to TinyImageNet, we observe a need for higher dimensions in both hidden and output layers. This suggests that datasets with more classes and larger sample sizes require higher-dimensional feature spaces for effective representation.
(c) Extremely low dimensions for the target output (e.g., 32) consistently yield the lowest performance across all datasets, indicating a minimum threshold for effective feature representation. This underfitting scenario likely occurs because the low-dimensional space is insufficient to separate different classes effectively.
(d) Conversely, very high dimensions can lead to overfitting, particularly in datasets with limited samples or high noise levels. This creates a delicate balance between representational power and generalization ability, which becomes more critical as dataset complexity increases.

The observed dimension-dependent behavior can be attributed to the role of the projector's output in our algorithm. As we store and refine these outputs over time, they effectively form a feature space for distinguishing different images. A larger feature space increases the model's capacity but also the risk of overfitting, particularly in scenarios with limited or noisy data.

Our findings highlight the importance of careful projector dimension tuning in DELA, especially when adapting the algorithm to new datasets or domains. They also suggest that adaptive dimension selection strategies could be a promising direction for future research, potentially allowing the algorithm to automatically adjust its projector dimensions based on the characteristics of the dataset at hand.

Impact of Randomly Initialized Models on Target Generation An intriguing aspect of our method is the use of randomly initialized models for target generation. Our investigations reveal that targets generated by these untrained models consistently outperform those created using standard normal distributions, suggesting that even without training, these models encapsulate certain priors about image content. To further explore this phenomenon, we conducted a comprehensive study

Table 9: **Comparison of targets generated by different randomly initialized model.** We analyze the impact of using targets generated by various randomly initialized model architectures on the performance of DELA. Results are reported in terms of test accuracy (%) on CIFAR-10, CIFAR-100, and TinyImageNet datasets. Random target serves as a baseline for comparison.

Model Architecture	CIFAR-10	CIFAR-100	TinyImageNet
Random Target	78.4	44.0	36.4
AlexNet	84.8	59.6	44.2
ResNet-18	83.2	57.1	43.0
ResNet-50	81.7	53.0	40.6
ResNet-101	78.9	45.3	39.7
DenseNet-121	83.6	57.6	43.1
DenseNet-161	84.1	58.2	43.1
DenseNet-169	83.9	57.6	43.8
MobileNet-v3-Small	85.1	59.0	44.0
MobileNet-v3-Large	84.8	59.2	44.0
EfficientNet-v2-S	84.1	57.7	42.9
EfficientNet-v2-M	82.8	57.0	42.5
EfficientNet-v2-L	81.9	54.7	42.0
ConvNeXt-v2-Small	85.4	**60.0**	44.4
ConvNeXt-v2-Base	**85.6**	**60.0**	44.4
ConvNeXt-v2-Large	85.5	59.9	44.2
ViT-B/16	84.0	58.7	**44.9**
ViT-B/32	83.4	58.4	44.2

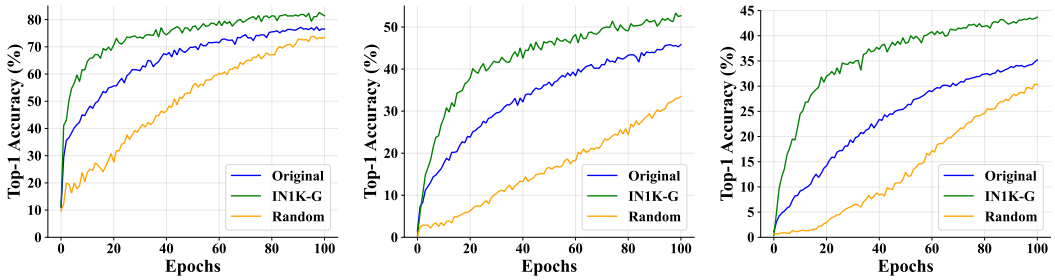
comparing the quality of targets generated by various randomly initialized architectures and analyzed their impact on convergence dynamics.

Table 9 presents the performance of DELA using targets generated by different model architectures on CIFAR-10, CIFAR-100, and TinyImageNet datasets. Our analysis yields several noteworthy observations:

(a) Across all tested architectures, randomly initialized model-generated targets significantly outperform purely random targets, demonstrating the inherent value of neural network structures in capturing image priors.
(b) Among the tested architectures, ConvNeXt-v2 variants consistently produced the highest quality targets for CIFAR-10 and CIFAR-100, leading to notable improvements in final accuracy compared to random targets. For TinyImageNet, the ViT-B/16 architecture performed best, achieving the most substantial improvement over random targets.
(c) Interestingly, within the same model family, smaller variants often outperformed their larger counterparts. This trend suggests that more compact architectures might better capture general image structures without overfitting to specific patterns, making them more suitable for generating diverse targets.
(d) Examining the performance trends chronologically, we observe that more recent architectures generally produce better targets. This pattern indicates that modern architectural designs, including carefully chosen parameters and module structures, may be more adept at capturing inherent data characteristics, even without training.
(e) The superior performance of certain architectures, such as ConvNeXt-v2 and ViT, in generating high-quality targets provides insights into the design principles that lead to better visual priors. This finding has implications not only for target generation in our method but also for the broader field of neural network architecture design and initialization strategies.

To further elucidate the impact of different target generation methods on both convergence dynamics and feature space distribution, we conducted additional analyses. Figure 7 illustrates the convergence behavior of DELA across different datasets using three distinct target generation approaches: our proposed method using a randomly initialized ConvNeXt-v2-Base model (Original), targets generated by an ImageNet-1K pre-trained model (IN1K-G) as a strong prior reference, and targets drawn from a standard normal distribution (Random) as a baseline.

The convergence analysis reveals several key insights:



(a) CIFAR-10 (b) CIFAR-100 (c) TinyImageNet

Figure 7: **Convergence analysis of DELA with different prior models across datasets.** This figure illustrates the accuracy convergence of DELA when initialized with three distinct methods for generating targets: Original (our default strategy using a randomly initialized ConvNeXt-v2-Base model), IN1K-G (targets generated by an ImageNet-1K pre-trained model), and Random (targets drawn from a standard normal distribution). Each subplot corresponds to a specific evaluation dataset.

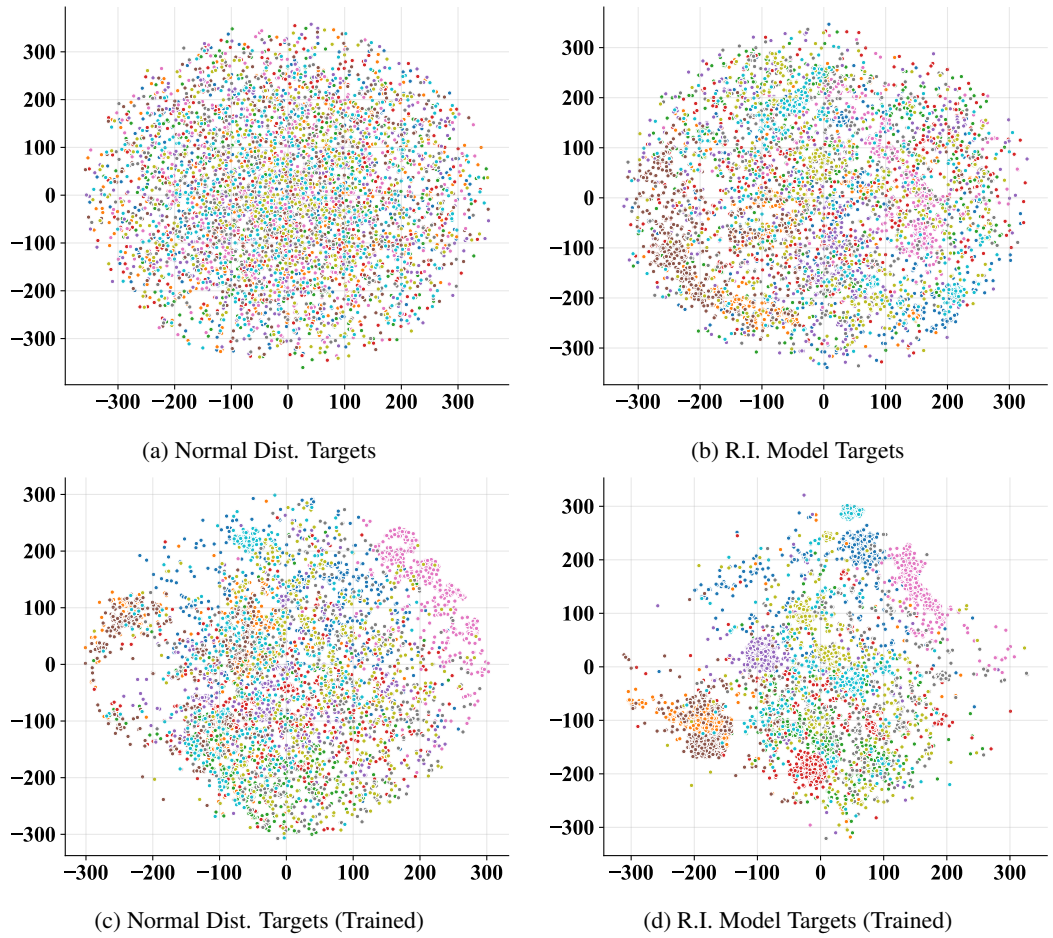
(a) Across all datasets, the convergence rate follows the order: IN1K-G > Original > Random, demonstrating the effectiveness of our proposed method in accelerating training compared to random initialization.
(b) The Random target approach exhibits high volatility in performance, particularly in the early stages of training, whereas both Original and IN1K-G show more stable learning curves.
(c) For larger datasets like TinyImageNet, the Random approach exhibits a slower, almost linear increase in performance over time. In contrast, both Original and IN1K-G demonstrate a more rapid initial improvement followed by a gradual plateau, indicative of more efficient learning dynamics. This accelerated learning curve suggests that these methods enable the model to quickly capture relevant features in the early stages of training, leading to faster convergence and better overall performance.
(d) The performance gap between our Original method and the strong IN1K-G prior narrows as training progresses, suggesting that our approach can achieve comparable results without relying on pre-trained models.

To complement our convergence analysis, we visualized the distribution of targets in the feature space at the beginning of training and after convergence using t-SNE dimensionality reduction. Figure 8 presents this comparison for targets generated using a standard normal distribution and our best-performing randomly initialized model (ConvNeXt-v2-Base).

This visualization reveals several additional insights:

(a) The initial distribution of model-generated targets exhibits a more structured and clustered arrangement compared to the uniform spread of normally distributed targets, suggesting that the randomly initialized model inherently captures some latent structure in the data space, even before training.
(b) Post-convergence, the model-generated targets demonstrate significantly tighter and more distinct clusters, indicating a more effective differentiation between classes and potentially contributing to the improved classification performance observed in our experiments.
(c) Targets derived from the normal distribution, while showing some clustering after training, exhibit less defined boundaries between classes, aligning with the performance gap noted in our quantitative results and underscoring the advantages of using model-generated targets.

These findings collectively underscore the critical role of target initialization in the overall efficacy of DELA. The choice of architecture for target generation significantly impacts the final performance of our method by influencing the initial distribution of targets, which in turn affects the optimization process. The effectiveness of randomly initialized models in producing high-quality targets suggests that these architectures inherently encode meaningful priors about image content, aligning with recent findings in the field of neural network initialization and architecture design (Ramanujan et al., 2020; Frankle & Carbin, 2018; Ulyanov et al., 2018). This observation not only highlights the importance of carefully selecting and designing target generation architectures but also opens new avenues for research in self-supervised learning and model initialization strategies. Our results demonstrate that by leveraging the inherent structural biases of neural networks, even before training,



(a) Normal Dist. Targets

(b) R.I. Model Targets

(c) Normal Dist. Targets (Trained)

(d) R.I. Model Targets (Trained)

Figure 8: **Feature space visualization of target distributions.** t-SNE-reduced representations of targets for CIFAR-100 dataset. (a) and (b) show the initial distribution of targets generated by a standard normal distribution and a randomly initialized ConvNeXt-v2-Base model, respectively. (c) and (d) depict the corresponding target distributions after training convergence.

Table 10: **Evaluating our DELA against self-supervised learning methods.** We analyze our DELA by training and evaluating models over unlabeled datasets, against seven conventional self-supervised learning methods.

Dataset	Architecture	Barlow	BYOL	DINO	MoCo	SimCLR	DCL	NNCLR	DELA
CF-10	ResNet-18	72.1 ± 0.1	72.8 ± 0.1	71.5 ± 0.0	72.5 ± 0.1	72.5 ± 0.1	72.3 ± 0.2	72.0 ± 0.3	**72.9 ± 0.0**
	ResNet-50	72.5 ± 0.0	72.3 ± 0.1	71.6 ± 0.1	72.7 ± 0.4	72.8 ± 0.3	72.8 ± 0.1	71.6 ± 0.2	**73.2 ± 0.4**
CF-100	ResNet-18	72.4 ± 0.1	72.7 ± 0.1	71.6 ± 0.1	72.7 ± 0.1	72.5 ± 0.2	73.0 ± 0.1	71.5 ± 0.2	**73.1 ± 0.0**
	ResNet-50	72.7 ± 0.5	72.3 ± 0.2	71.6 ± 0.3	72.9 ± 0.1	72.6 ± 0.6	**73.0 ± 0.1**	71.0 ± 0.1	**73.0 ± 0.3**
T-IN	ResNet-18	74.1 ± 0.1	**74.2 ± 0.1**	72.7 ± 0.2	73.7 ± 0.1	73.8 ± 0.2	73.8 ± 0.1	72.8 ± 0.1	74.0 ± 0.1
	ResNet-50	**74.4 ± 0.2**	73.8 ± 0.3	71.8 ± 1.4	73.6 ± 0.1	73.2 ± 0.1	73.5 ± 0.2	72.1 ± 0.1	73.9 ± 0.3
IN-1K	ResNet-50	78.3 ± 0.3	79.1 ± 0.7	78.7 ± 0.2	79.4 ± 0.6	80.1 ± 0.2	79.8 ± 0.2	79.7 ± 0.3	**80.3 ± 0.1**

we can significantly enhance the performance and efficiency of self-supervised learning algorithms, potentially leading to more robust and effective representation learning across various domains.

E ADDITIONAL RESULTS

We evaluate different self-supervised learning methods and our DELA on semantic segmentation using the VOC 2012 dataset (Everingham et al., 2015). Semantic segmentation involves classifying each pixel of an image. The experimental results are shown in Table 10, which demonstrate our

Table 11: **Comparison of model performance using original noisy data and evolved data.** We report the test accuracy (%) on MNIST (MT), Fashion-MNIST (F-MT), and CIFAR-10 (CF-10) datasets with varying levels of label noise. 'Original' denotes training with the original noisy dataset, while 'RN18-E' and 'RN50-E' represent training with evolved data obtained using ResNet18 and ResNet50, respectively.

Dataset	Arch.	Noise Rate 0.2			Noise Rate 0.4			Noise Rate 0.6		
		Original	RN18-E	RN50-E	Original	RN18-E	RN50-E	Original	RN18-E	RN50-E
MT	RN18	**99.3**	99.2	99.2	99.2	**99.3**	99.1	**99.3**	99.2	99.1
	RN50	**99.4**	99.3	**99.4**	**99.4**	99.3	99.1	99.4	**99.5**	99.4
F-MT	RN18	**92.1**	91.5	91.0	**91.9**	89.5	90.5	91.2	**91.8**	89.5
	RN50	**93.6**	92.0	90.5	**93.4**	90.0	89.4	**93.4**	92.5	91.8
CF-10	RN18	85.3	**88.5**	86.5	84.5	**86.5**	84.3	82.7	**87.0**	84.5
	RN50	87.5	**91.0**	90.7	86.0	**90.5**	90.2	83.3	**90.1**	89.8

DELA demonstrates strong performance and generalization capabilities, excelling especially on larger datasets like ImageNet-1K.

To evaluate the robustness of DELA against noisy labels, we conducted experiments on MNIST, Fashion-MNIST, and CIFAR-10 datasets with varying levels of label noise (20%, 40%, and 60%). Table 11 presents the test accuracies achieved using the original noisy datasets and the evolved datasets generated by DELA with ResNet18 and ResNet50 architectures. The results demonstrate that DELA effectively "purifies" the noisy datasets, leading to improved model accuracy, particularly for CIFAR-10. Notably, the performance of models trained on evolved data remains relatively stable across different noise levels, especially for CIFAR-10, where DELA consistently outperforms training on original noisy data. This stability underscores the robust nature of our method in handling noisy data.