FROM DETRIMENTAL TO BENEFICIAL: DYNAMIC INFLUENCE-BASED VALUATION AND EDITING

Anonymous authors
Paper under double-blind review

ABSTRACT

Data valuation is a cornerstone of data-centric learning, where prior efforts primarily focus on designing algorithms to classify training samples as either beneficial or detrimental for the learning task. However, leveraging these valuation estimates for subsequent data intervention remains underexplored; conventional approaches typically discard or downweight harmful samples, thereby underutilizing available data resources. In this paper, we present Dynamic Influence-based Valuation and Editing (DIVE), a novel and efficient framework that dynamically estimates sample values at the batch level and transforms detrimental data into beneficial contributions. Rather than altering the raw data, DIVE operates at the optimization level by strategically reversing the gradient directions of harmful samples during training, ensuring seamless integration with standard learning procedures with minimal overhead. Extensive empirical evaluations demonstrate that DIVE consistently improves classification performance, maximizes data efficiency, stabilizes optimization, and effectively generalizes to large language model fine-tuning.

1 INTRODUCTION

Recent years have witnessed growing interest in data quality control, giving rise to the field of data-centric learning (Koh & Liang, 2017; Hammoudeh & Lowd, 2024; Jia et al., 2018). Unlike model-centric approaches, which design novel algorithms with fixed training data, data-centric learning focuses on curating data to better suit the learning algorithm and improve performance. In contrast to conventional preprocessing techniques—such as normalization or outlier removal—which are often decoupled from downstream tasks, data-centric learning explicitly evaluates the contribution of data with respect to the target model. This enables more targeted and interpretable interventions across a range of applications, including training subset selection, synthetic data generation, noisy label detection, active learning, and fairness improvement.

Data influence estimation, a core component of data-centric learning, can be broadly categorized into two groups. Retraining-based methods evaluate the influence of a sample by retraining the model with and without it and measuring the resulting performance change, including classical leave-one-out approaches (Cook & Weisberg, 1982) and Shapley value–based methods (Ghorbani & Zou, 2019; Jia et al., 2019; Kwon & Zou, 2022; Jia et al., 2018). In contrast, gradient-based methods estimate influence without retraining, most notably through influence functions. While avoiding the cost of retraining, these methods introduce a new computational challenge in approximating the inverse Hessian. The seminal work of (Koh & Liang, 2017) addresses this via a Taylor expansion and LiSSA (Agarwal et al., 2017) for efficient approximation. Subsequent work further improves scalability through stochastic recursive methods (Koh & Liang, 2017), random projections (Schioppa et al., 2022), Kronecker-factored eigendecomposition (Grosse et al., 2023), and low-rank approximations (Park et al., 2023; Kwon et al., 2023). More recently, Hessian-free approaches simplify computation by approximating the inverse Hessian with the identity matrix (Charpiat et al., 2019; Pruthi et al., 2020; Yang et al., 2026). Beyond static estimation, dynamic data valuation has recently attracted increasing attention (Wang et al., 2024; Yang et al., 2025).

Contributions. While data-centric learning has developed rapidly, most existing work focuses on designing efficient and accurate data valuation estimators, with limited attention to how these estimates are subsequently utilized. In particular, prior methods typically remove or downweight identified detrimental samples, thereby discarding potentially useful information. In contrast, we aim

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

to fully exploit these detrimental samples by transforming them into beneficial contributions, leading to improved learning performance. We summarize our main contributions as follows:

- *Concept*: To the best of our knowledge, we are the first to transform detrimental samples into beneficial contributions, making every sample count during learning. This perspective improves data efficiency by leveraging all available samples, rather than discarding potentially useful information. It also leads to more stable and effective optimization by mitigating harmful gradient signals during training.
- *Technique*: We propose Dynamic Influence-based Valuation and Editing (DIVE), a simple and efficient framework that dynamically estimates data value at the batch level and modifies the gradients of detrimental samples to make them beneficial for model updates. Notably, we do not alter the raw data; instead, we operate at the gradient level, enabling seamless integration into standard optimization procedures with minimal overhead.
- *Evaluation*: We conduct extensive experiments to demonstrate the effectiveness of DIVE, comparing it against a range of data valuation and dynamic batch curation methods. We further provide diverse in-depth analyses and extend our approach to large language model fine-tuning.

2 Related Work

In this section, we introduce two most related directions, influence functions and dynamic batch curation, and then highlight the key differences between our paper and the literature.

Influence Function. Influence functions, a mainstream tool for data valuation, were originally developed in robust statistics (Hampel, 1974; Cook & Weisberg, 1982; Martin & Yohai, 1986) to quantify the sensitivity of model parameters to perturbations in training data. They were introduced to the machine learning community by (Koh & Liang, 2017) and have since been widely used to estimate sample influence with respect to validation performance.

To address the high computational cost of influence functions, several works propose efficient approximations of the inverse Hessian. LiSSA (Koh & Liang, 2017) leverages Hessian–vector products for stochastic approximation, while EKFAC (Grosse et al., 2023) exploits structured eigenvalue decompositions. DataInf (Kwon et al., 2023) further simplifies influence estimation by leveraging a rank-1 structure of the empirical Hessian, and TRAK (Park et al., 2023) employs random projections to obtain a tractable kernel-based approximation. More recently, Hessian-free influence methods have been proposed, which approximate the inverse Hessian with an identity matrix for scalability (Charpiat et al., 2019; Pruthi et al., 2020; Yang et al., 2026; Killamsetty et al., 2021).

Beyond classical influence formulations, several recent works revisit influence estimation from an optimization-aware and non-convex perspective. SGD-Influence (Hara et al., 2019) tracks the stochastic gradient descent trajectory and estimates the effect of removing a training sample by reusing intermediate iterates, thereby relaxing the local convexity assumption. MoSo (Tan et al., 2023) models data pruning by approximating a moving-one-sample-out update along the optimization path, while Z0-Inf (Kokhlikyan et al., 2025) develops a zeroth-order estimator that avoids explicit gradients and Hessian computations.

Beyond static estimation, dynamic data valuation has recently attracted increasing attention (Wang et al., 2024; Yang et al., 2025). Early approaches approximate dynamic influence by aggregating influence estimates across training checkpoints (Pruthi et al., 2020; Grosse et al., 2023; Park et al., 2023). However, such aggregation fails to capture the evolving nature of influence and may cancel out conflicting signals. More recent work directly estimates data value at each training step, enabling truly dynamic valuation (Wang et al., 2024; Yang et al., 2025).

Dynamic Batch Curation. The dynamic batch curation paradigm is conceptually related to several research directions. Curriculum learning (Bengio et al., 2009; Wang et al., 2021; Soviany et al., 2022) introduces a structured training schedule by presenting samples in an easy-to-hard order, mimicking human learning processes. Another line focuses on adaptive data selection or reweighting during training to improve model performance. For example, reducible holdout loss selection and its variants (Mindermann et al., 2022) prioritize samples based on difficulty and uncertainty, while JEST (Evans et al., 2024) performs joint multimodal selection to improve training efficiency. ACID/ACED (Udandarao et al., 2025) further targets efficient distillation through adaptive data

selection. A related direction adjusts sampling probabilities based on gradient information. GradientIS and GradientMIS (Salaün et al., 2025b;a) reweight samples according to the norm of their output-layer gradients to reduce stochastic gradient variance and accelerate optimization. More broadly, several recent works aim to improve efficiency by pruning less informative samples based on estimated scores (Toneva et al., 2018; Raju et al., 2021; He et al., 2024; Qin et al., 2023).

While techniques like Gradient Reversal (Ganin & Lempitsky, 2015; Ganin et al., 2016) or Negative Learning (Kim et al., 2019; 2021) also adjust gradient directions for unlearning or robust training, they inherently rely on external, prior knowledge or heuristics to pre-identify noisy/detrimental samples. In contrast, DIVE integrates sample valuation and gradient editing into a unified data-centric framework. It dynamically discovers which samples are detrimental without any explicit prior labels or loss-threshold heuristics, automatically turning harmful gradients into beneficial updates on the fly.

To the best of our knowledge, existing data-centric learning approaches primarily focus on identifying and removing or downweighting detrimental samples. In contrast, our work takes a fundamentally different perspective by transforming detrimental samples into beneficial contributions at the optimization level. At the technical level, since data valuation evolves during model optimization, we adopt a dynamic formulation that estimates data valuation at each batch. Based on this, we edit detrimental samples at the gradient level, rather than modifying the raw data, enabling seamless integration into standard optimization procedures.

3 Preliminaries on Influence Functions

Consider a classifier with parameters $\theta \in \mathbb{R}^D$ mapping instances $z=\{x, y\}$ from input space $x \in \mathcal{X}$ to output space $y \in \mathcal{Y}$, the model parameters $\hat{\theta}=\text{argmin}_{\theta\in\Theta} \frac{1}{n} \sum_{i=1}^{n} \ell(z_i;\theta)$ can be obtained by solving the empirical risk minimization problem. If we downweight a training sample z_j by a very small fraction ϵ, the new parameters can be $\hat{\theta}(z_j;-\epsilon) = \text{argmin}_{\theta\in\Theta} \frac{1}{n}(\sum_{i=1}^{n} \ell(z_i;\theta) - \epsilon\ell(z_j;\theta))$. By evaluating the limit as ϵ approaches 0, the seminal work of Koh & Liang (2017) provides an estimation for the influence score associated with the removal of z_j from the training set:

$$\mathcal{I}(z_j;\hat{\theta}) = \sum\nolimits_{z\in\mathcal{V}} \nabla\ell(z;\hat{\theta})^\top \mathbf{H}_{\hat{\theta}}^{-1} \nabla\ell(z_j;\hat{\theta}), \tag{1}$$

where $\mathcal{V}$ denotes the validation set (self-influence employs the training set instead), $\nabla\ell(z_j;\hat{\theta})$ is the gradient of sample z_j, and $\mathbf{H}_{\hat{\theta}}=\sum_{i=1}^{n} \nabla^2\ell(z_i;\hat{\theta})$ denotes the Hessian matrix.

It is well recognized that influence functions rely on several strong assumptions, including model convexity and access to a fully converged solution. While many studies have extended influence functions to deep models with promising empirical results (Chhabra et al., 2024), they primarily focus on improving computational efficiency, particularly in approximating the inverse Hessian matrix. Common strategies include stochastic recursive approximation (Koh & Liang, 2017), random projection (Schioppa et al., 2022), Kronecker-Factored eigendecomposition (Grosse et al., 2023), low-rank or rank-1 approximations (Park et al., 2023; Kwon et al., 2023), and simplifying the inverse Hessian as the identity matrix, leading to Hessian-free influence functions (Charpiat et al., 2019; Pruthi et al., 2020; Yang et al., 2026).

4 Method

In this section, we introduce our Dynamic Influence-based Valuation and Editing (DIVE), a simple and efficient method that estimates the data valuation in the dynamic batch and modifies the gradient of the detrimental samples to be beneficial ones for classification model updating.

Dynamic Data Valuation. Influence functions are widely used for data valuation, estimating the impact of individual training samples on a target objective, typically the validation loss. Most existing influence function–based methods compute data values using a converged model and then apply these estimates to guide subsequent actions, such as removing detrimental samples and retraining the model on the modified dataset. However, this procedure has two key limitations. First, it produces static data valuations, as the influence scores are derived from a fixed, converged model. As a result, samples identified as detrimental may not remain harmful once the training set is modified and the model is retrained. Second, the approach requires two rounds of training: one to obtain the converged

model for valuation and another to retrain on the filtered dataset. This makes it computationally prohibitive for large-scale datasets and models.

Recently, dynamic data valuation has gained increasing attention (Wang et al., 2024; Yang et al., 2025). Instead of estimating data value based on a fully converged model, these approaches compute data valuation at the mini-batch level during training and use it to guide optimization. While this paradigm addresses the limitations of static valuation—namely, outdated estimates and the need for retraining—it introduces significant computational overhead due to the frequent evaluation of data values. As a result, only highly efficient approximations, such as Hessian-free influence functions, are practical in large-scale settings.

In this paper, we adopt a dynamic data valuation framework. Let $\mathcal{B} = \{z_j\}_{j=1}^{b}$ denote a mini-batch of size b. The influence of a sample z_j at training step t is defined as:

$$\mathcal{I}(z_j; \theta^{(t)}) = \sum\nolimits_{z \in \mathcal{V}} \nabla \ell(z; \theta^{(t)})^\top \cdot \nabla \ell(z_j; \theta^{(t)}), \tag{2}$$

where $\theta^{(t)}$ is the model parameter at t-th epoch. Eq. equation 2 can be interpreted as a Hessian-free approximation of influence functions that avoids the strong assumptions required by conventional formulations. Intuitively, model updates driven by training samples that are aligned with the validation objective will tend to reduce validation loss. This alignment is not defined in the original feature space, but in the gradient space, which directly governs parameter updates. As a result, Eq. equation 2 does not rely on assumptions such as model convexity or access to a fully converged solution. Its effectiveness has been empirically validated in prior work (Wang et al., 2024) and (Yang et al., 2025). However, these studies are implemented in PyTorch, which only provides batch-level gradients by default, whereas Eq. equation 2 requires sample-level gradients. To address this limitation, they introduce approximations such as ghost influence and layer-wise estimation.

In contrast, we implement our method in JAX, which natively supports efficient computation of sample-level gradients, enabling direct and accurate estimation of Eq. equation 2. Based on the sign of $\mathcal{I}(z_j; \theta^{(t)})$, the batch $\mathcal{B}$ can be partitioned into $\mathcal{B}+$ and $\mathcal{B}-$, corresponding to samples with positive and negative influence, respectively, for the following editing.

Dynamic Gradient Editing. Unlike existing dynamic data valuation methods (Wang et al., 2024; Yang et al., 2025), which discard detrimental samples at the batch level, we instead transform such samples into beneficial ones. As shown in Eq. equation 2, the valuation is given by the inner product between gradients, which determines whether a sample contributes positively or negatively to the target objective. For samples with negative influence, we flip the direction of their gradients to convert them into beneficial updates.

Importantly, our approach does not modify the raw data, but directly edits the gradients, allowing seamless integration into standard optimization procedures. The parameter update is defined as:

$$\theta^{(t+1)} = \theta^{(t)} - \eta \left(\sum_{z_j \in \mathcal{B}+} \nabla \ell(z_j; \theta^{(t)}) - \sum_{z_j \in \mathcal{B}-} \nabla \ell(z_j; \theta^{(t)}) \right), \tag{3}$$

where η is the learning rate. In this way, every sample contributes constructively to the training process, improving data efficiency in dynamic learning.

5 Experimental Results

In this section, we first introduce the experimental setup, including datasets, baselines, and implementation details. We then evaluate the performance of DIVE in terms of both effectiveness and efficiency by comparing it with existing baseline methods, followed by in-depth analyses to better understand its behavior. Finally, we extend DIVE to large-scale LLM fine-tuning.

5.1 Experimental Setup

Datasets. We evaluate our method on six noisy-label image benchmarks. *CIFAR-10N* (50,000 training images, 10 classes) provides three human-annotated noise settings—*aggregate* ($\sim$9%), *random* ($\sim$17%), and *worst* ($\sim$40%)—and *CIFAR-100N* (50,000 images, 100 classes) carries $\sim$40%

Table 1: Accuracy of data-centric methods on noisy benchmarks with a validation set

Method	*CIFAR-10Na*	*CIFAR-10Nw*	*CIFAR-10Nr*	*CIFAR-100N*	*Animal-10N*	*Food-101N*	Avg.
Vanilla	88.94±0.16	78.97±0.22	86.62±0.28	54.41±0.49	82.60±0.26	72.42±0.14	77.33
LiSSA (2017)	89.03±0.50	78.38±0.50	86.32±0.13	54.67±0.30	82.41±0.25	72.07±0.22	77.15
IP (2026)	89.54±0.19	77.82±0.40	86.54±0.11	55.03±0.18	82.72±0.35	72.13±0.27	77.30
SPL (2010)	88.98±0.26	70.53±0.85	86.32±0.34	38.47±0.55	83.38±0.18	**72.97±0.07**	73.44
InfoBatch (2023)	89.64±0.17	80.90±0.34	87.92±0.21	56.36±0.47	83.52±0.30	72.83±0.09	78.53
GHOST (2024)	**90.11±0.16**	83.25±0.26	88.74±0.17	**57.52±0.43**	84.10±0.38	72.73±0.16	79.41
LAI (2025)	90.10±0.12	82.98±0.25	88.63±0.13	**57.52±0.47**	84.32±0.19	72.64±0.20	79.36
DIVE$_r$ (Ours)	89.69±0.05	81.54±0.31	88.11±0.16	56.84±0.43	83.44±0.25	72.59±0.18	78.70
DIVE (Ours)	90.00±0.16	**83.48±0.18**	**88.77±0.22**	57.38±0.52	**84.53±0.17**	72.91±0.26	**79.51**

label noise (Wei et al., 2022). *Animal-10N* (50,000 training / 5,000 test images, 10 classes) contains $\sim$8% human-annotation noise (Song et al., 2019), and *Food-101N* (75,750 training / 25,250 test images, 101 classes) carries $\sim$20% intrinsic web-label noise in its training split (Bossard et al., 2014). The noise level thus ranges from $\sim$8% to $\sim$40%. We consider two reference settings: (i) *with a validation set*, where influence is estimated against a small clean reference (1,000 clean-labeled held-out samples; for Animal-10N and Food-101N, which have no clean training labels, we set aside a class-stratified 1/5 of the clean test split as validation and evaluate on the remaining 4/5), and (ii) *self-influence*, which requires no validation data and uses the mini-batch mean gradient as the reference. The details of datasets and models can be found in Appendix A.

Baselines. We compare against seven baselines across four categories: (1) *Vanilla*: standard SGD; (2) *Static influence*: LiSSA (Koh & Liang, 2017) and IP (Yang et al., 2026); (3) *Curriculum*: SPL (Kumar et al., 2010); (4) *Dynamic*: InfoBatch (Qin et al., 2023), Ghost (Wang et al., 2024), LAI (Yang et al., 2025), our DIVE$_r$ (removal variant), and our DIVE.

Implementation. For all image datasets we use a ResNet-9 (He et al., 2016) backbone with a three-layer MLP head; sample influence is scored only on the head using cached penultimate features, so the convolutional backbone is never differentiated per sample. Models are trained with SGD (Nesterov momentum 0.9, decoupled weight decay 5×10^{-4}, batch size 512, standard augmentation) under a one-cycle learning-rate schedule with peak learning rate 0.1 (linear warm-up over the first 15% of steps, then annealing to $\sim$0). We train for 30 epochs on *CIFAR-10N* and *CIFAR-100N*, 40 on *Animal-10N*, and 100 on *Food-101N*, applying gradient clipping (global norm 1.0) on the higher-resolution *Animal-10N* and *Food-101N*. DIVE$_r$ and DIVE use a warm-up phase of 30% of the training epochs, editing every 2 mini-batches with last-layer gradients. All experiments are implemented in JAX and executed on Google Cloud TPU v4-8. We report the best test accuracy as mean $\pm$ standard deviation over 4 seeds.

5.2 Algorithmic Performance

To evaluate the effectiveness of DIVE, we consider the task of classification with noisy labels. Table 1 reports the average classification accuracy and standard deviation over five runs on six noisy benchmark datasets. Overall, DIVE consistently achieves the best performance on three out of six datasets and attains the highest average accuracy of 0.7951, demonstrating strong robustness to label noise across diverse tasks. Notably, DIVE significantly outperforms the second-best method in terms of average performance under a paired t-test ($p < 0.1$).

Compared with static influence-based methods (e.g., LiSSA and IP), DIVE consistently achieves substantial improvements across all datasets, demonstrating the benefits of dynamic influence estimation over static approximations. DIVE also consistently outperforms curriculum-based learning (SPL), suggesting that simple curriculum heuristics are ineffective under severe label noise. In fact, SPL even underperforms the vanilla baseline. Among dynamic approaches, InfoBatch, Ghost, and LAI are all competitive and consistently outperform static methods; nevertheless, DIVE further achieves consistent gains over these strong baselines.

Beyond the above high-level observations, several additional insights can be drawn. (1) Ghost, LAI, DIVE$_r$, and DIVE all belong to the same dynamic data curation framework and employ the same first-order influence estimation, differing only in how identified detrimental samples are handled. Ghost and LAI remove detrimental samples at every training batch, whereas DIVE$_r$ performs removal less frequently (once every second mini-batch). This less aggressive intervention substantially

Table 2: Training overhead relative to the vanilla baseline.

Method	*CIFAR-10Na*	*CIFAR-10Nw*	*CIFAR-10Nr*	*CIFAR-100N*	*Animal-10N*	*Food-101N*
Vanilla	1.00	1.00	1.00	1.00	1.00	1.00
LiSSA (2017)	2.50	2.65	2.64	2.62	2.80	2.50
IP (2026)	2.50	2.64	2.62	2.65	2.80	2.51
SPL (2010)	1.00	1.00	1.00	1.01	1.01	1.00
InfoBatch (2023)	1.01	1.02	1.02	1.01	1.02	1.01
GHOST (2024)	1.22	1.22	1.22	1.22	1.25	1.22
LAI (2025)	1.18	1.18	1.19	1.19	1.24	1.21
DIVE$_r$ (Ours)	1.09	1.09	1.09	1.09	1.12	1.11
DIVE (Ours)	1.09	1.09	1.09	1.09	1.12	1.11

reduces the computational overhead while maintaining competitive performance, suggesting that overly frequent data curation is unnecessary. (2) DIVE further extends DIVE$_r$ by transforming detrimental samples into beneficial ones instead of simply removing them. As shown in Table 1, DIVE consistently outperforms DIVE$_r$ across all datasets, with statistically significant improvements. This suggests that transforming detrimental samples better preserves informative yet noisy training examples, allowing the model to exploit their underlying signal rather than discarding potentially useful information. Consequently, DIVE achieves more stable optimization and better generalization under noisy supervision.

To complement the above analysis, we further examine the computational efficiency of different data curation methods. Table 2 reports the training overhead relative to the vanilla baseline. LiSSA and IP require two rounds of training and therefore incur the highest computational overhead. SPL follows a pre-defined curriculum and incurs only marginal runtime overhead. Among dynamic methods, InfoBatch performs data curation at every batch while remaining relatively efficient, as it relies only on forward losses without computing sample-level gradients. However, as shown in Table 1, this efficiency comes at the cost of inferior accuracy compared with gradient-based approaches. Ghost, LAI, DIVE$_r$, and DIVE all rely on sample-level gradients. Compared with Ghost and LAI, DIVE incurs lower computational overhead by performing detrimental sample editing once every other epoch instead of at every mini-batch. This less frequent intervention not only substantially improves efficiency but also avoids excessive interference with the optimization process, leading to better overall performance. Finally, although DIVE further transforms detrimental samples into beneficial ones, this operation introduces virtually no additional computational overhead compared with DIVE$_r$.

5.3 In-depth Exploration

Beyond reporting the algorithmic performance in terms of effectiveness and efficiency in the above, we continue to provide diverse in-depth explorations of our DIVE as follows.

Detrimental Sample Verification. In the data valuation literature, sample valuation is typically evaluated by measuring the correlation between the estimated values and the leave-one-out loss changes (Koh & Liang, 2017; Wang et al., 2024). In contrast, our work focuses on identifying detrimental samples, rather than precisely estimating their exact values. Although the influence estimation in Eq. equation 2 provides only a coarse approximation, it still preserves the polarity of samples—i.e., whether they are beneficial or detrimental. Based on this observation, we adopt a more direct evaluation criterion: a sample is considered correctly identified as detrimental if correcting it leads to an improvement in model performance. This criterion better aligns with our goal of effective detrimental sample identification, rather than accurate correction.

Figure 1A illustrates the test performance of several dynamic data curation methods and the vanilla baseline on *CIFAR-10N-w* across training epochs. After the warm-up phase, DIVE consistently outperforms the vanilla method, with the performance gains primarily attributed to the correction of detrimental samples. Beyond verifying the effectiveness of detrimental sample correction, two additional observations can be made. (1) During the warm-up phase, the vanilla method achieves the best performance and exhibits a steady upward trend, suggesting that sample quantity is the dominant factor at this stage. As training progresses, the dominant factor shifts to sample quality, where improvements are achieved by either removing or correcting detrimental samples. (2) Correcting detrimental samples is a stronger operation than removing them: as reported in Table 3, DIVE$_r$ (removal) already improves over the vanilla baseline, yet DIVE attains a consistently higher accuracy

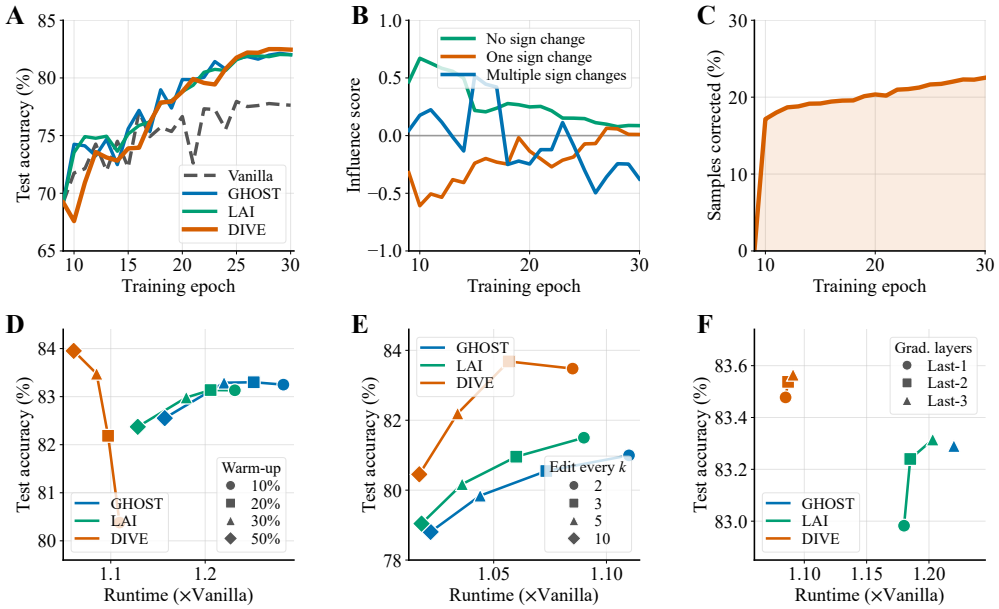

Figure 1: In-depth exploration of DIVE on *CIFAR-10N-w*. **(A)** Test accuracy vs. epoch, shown from the end of warm-up. **(B)** Influence scores of three individual training samples across epochs (from the end of warm-up). **(C)** Fraction of samples corrected by DIVE per epoch. In (D)–(F) the horizontal axis is training runtime relative to vanilla, colour denotes the method, and marker shape denotes the ablated hyperparameter; points of the same method are joined by a line. **(D)** Warm-up-duration ablation (marker: 10/20/30/50%). **(E)** Editing-frequency ablation (marker: edit every $k \in \{2, 3, 5, 10\}$ mini-batches). **(F)** Scoring-layer ablation (marker: last 1/2/3 gradients for DIVE and LAI; GHOST uses the full head, a single point).

(e.g., 82.67 vs. 80.98 on *CIFAR-10N-w* and 79.03 vs. 78.17 on average), demonstrating that the gains stem more from *correcting* detrimental samples than from merely discarding them.

Dynamic Mechanism. We further examine how DIVE acts on the training set as optimization proceeds. Figure 1B tracks the influence score of three representative training examples across epochs and shows that a sample's influence *sign* is not constant: the same sample can be beneficial (score above zero) at some epochs and detrimental (below zero) at others. The three examples differ only in how often this sign flips—one never changes sign, remaining beneficial throughout; a second changes sign once, starting detrimental and becoming beneficial as training proceeds; and a third changes sign multiple times, oscillating between beneficial and detrimental. Whether a sample helps or harms the model is therefore not an intrinsic, fixed property but depends on the current state of optimization. Figure 1C aggregates this effect over the whole training set, reporting the fraction of samples corrected by DIVE at each epoch: no editing occurs during the warm-up phase, after which the correction rate rises as the model fits the data and stabilizes at around 20%, remaining consistently non-zero for the remainder of training. Together these confirm that the set of detrimental samples is not static but continuously evolves as optimization proceeds, and thus an adaptive, per-step correction strategy is necessary rather than a one-shot static valuation.

Performance-Cost Analysis. We analyze the performance-cost of DIVE from three perspectives: the warm-up duration, the choice of gradient computation across different layers, and the frequency of detrimental sample correction.

Warm-up Duration. Figure 1D plots accuracy against runtime as the warm-up duration varies; a longer warm-up shortens the dynamic editing phase and therefore also lowers runtime. DIVE's accuracy keeps improving as warm-up lengthens (from 10% to 50% of epochs)—becoming both more accurate and cheaper—reflecting that a better-fit reference model yields more reliable influence estimates; we nonetheless adopt a 30% warm-up as the default, which already captures most of the gain and matches the budget used across all main experiments. In contrast, the removal-based GHOST/LAI peak early and then decline, since prolonged vanilla training leaves fewer clearly-detrimental samples for them to profitably discard.

Table 3: Accuracy of data-centric methods on noisy benchmarks without a validation set

Method	*CIFAR-10Na*	*CIFAR-10Nw*	*CIFAR-10Nr*	*CIFAR-100N*	*Animal-10N*	*Food-101N*	Avg.
Vanilla	88.94±0.16	78.97±0.22	86.62±0.28	54.41±0.49	82.60±0.26	72.42±0.14	77.33
LiSSA (2017)	88.73±0.15	78.38±0.50	86.23±0.22	54.45±0.15	82.41±0.25	72.00±0.10	77.03
IP (2026)	88.67±0.53	78.31±0.42	86.56±0.49	54.55±0.25	82.60±0.45	72.03±0.16	77.12
SPL (2010)	89.00±0.17	70.53±0.85	86.01±0.45	38.47±0.55	83.27±0.51	72.79±0.13	73.34
InfoBatch (2023)	89.74±0.06	81.12±0.24	88.02±0.19	56.42±0.35	83.29±0.40	**72.80±0.22**	78.56
GHOST (2024)	89.71±0.09	82.26±0.32	88.45±0.17	**55.96±0.30**	84.16±0.19	72.70±0.09	78.87
LAI (2025)	89.77±0.09	82.34±0.42	88.42±0.06	55.63±0.39	84.10±0.28	72.50±0.22	78.79
DIVE$_r$ (Ours)	89.52±0.11	80.98±0.33	87.78±0.18	55.02±0.59	83.13±0.25	72.57±0.21	78.17
DIVE (Ours)	**90.00±0.16**	**82.67±0.17**	**88.64±0.12**	55.66±0.37	**84.61±0.27**	72.61±0.20	**79.03**

Editing Frequency and Scoring Layers. We further study the frequency of detrimental sample editing, defined as editing once every k mini-batches during the dynamic phase; a larger k reduces the editing runtime. As shown in Figure 1E, DIVE's accuracy peaks at a small k (k=2 and 3) and then declines as k grows, yet it dominates GHOST and LAI at every runtime budget. Very frequent editing applies gradient ascent to the detrimental samples too aggressively—at the extreme of every mini-batch it destabilizes training and causes a sharp accuracy drop—whereas very infrequent editing (e.g., k=10) fails to correct the accumulated noise in time. A moderate frequency (k=2) achieves the best accuracy–runtime balance, which we adopt as the default. In contrast, the removal-based methods (GHOST, LAI) exhibit the *opposite* accuracy trend and monotonically favor more frequent editing, as discarding samples is gentler and does not destabilize optimization.

Scoring-layer ablation. Figure 1F ablates the number of gradient layers used for scoring: DIVE attains the highest accuracy at the lowest runtime and is insensitive to the number of last-k gradient layers, whereas LAI must include more gradient layers—raising its cost toward GHOST, which always scores over the full head—to improve. This indicates that DIVE's sign-correction remains effective even from a single last-layer gradient, keeping its scoring cost minimal.

Based on these observations, we use the last-layer gradient, a 30% warm-up duration, and editing every 2 mini-batches (k=2) as the default configuration. Similar phenomena on *Animal-10N* and *Food-101N* are shown in Appendix B.

Self-Influence. We further evaluate DIVE in the absence of a validation set by adopting self-influence, where training samples are used to estimate influence scores. Specifically, the first term in Eq. equation 2 is replaced by the average gradient of the training samples within each mini-batch. As shown in Table 3, DIVE consistently achieves the best average performance and significantly outperforms the second-best method ($p < 0.1$), demonstrating that our method remains effective even when no clean validation data are available.

5.4 Extension on Fine-Tuning LLMs

Beyond the vision benchmarks considered above, we further examine whether DIVE remains effective in the setting of LLM fine-tuning, where the base model is orders of magnitude larger, training is limited to a single epoch, and label noise is inherent to the datasets rather than synthetically injected. This regime differs from the image benchmarks in three important aspects. First, per-sample full-model gradients are infeasible—Gemma-3-12B has ∼12B parameters, so per-sample gradient tensors on a mini-batch would exceed device memory by orders of magnitude. Following the last-layer approximation identified in the performance-cost analysis, we restrict influence estimation to the LoRA adapters only. Second, training is short (one epoch) with a cosine-decayed learning-rate schedule and warm-up, so an aggressive intervention schedule is not appropriate; DIVE is invoked only after a fraction of the epoch is completed. Third, a validation split is not always available with a clean, distribution-matched reference (e.g., *Civil_Comments* follows a natural ∼8/92 toxic/non-toxic distribution), so we adopt the self-influence variant developed in the previous section, using the mean gradient of the current training mini-batch as the reference direction.

Experimental Setting. We fine-tune *Gemma-3-12B-IT* in a generative supervised-fine-tuning formulation: each classification example is verbalized as a prompt–response pair, and the loss is computed only on the response tokens. LoRA adapters (rank 16, $\alpha = 32$) are attached to all attention and MLP projections; all other parameters are frozen. Optimization uses Adam with peak learning rate 2×10^{-4}, a linear warm-up over the first few hundred steps followed by cosine decay to zero,

Table 4: Performance of DIVE for LLM fine-tuning

Dataset	#Class	#Train	#Test	Test Loss		Test Accuracy	
				Vanilla	DIVE	Vanilla	DIVE
PAWS	2	49,401	8,000	0.1584	**0.1549**	**0.9515**	0.9514
ANLI_r3	3	100,459	1,200	0.8049	**0.7374**	0.6806	**0.6988**
Civil_Comments_b	2	200,000	97,320	0.1366	**0.1150**	0.8377	**0.8590**
Civil_Comments	2	1,804,874	97,320	0.0702	**0.0676**	**0.9531**	0.9518

effective batch size 16, and one epoch over the training set. The DIVE schedule mirrors the CV configuration: the first 30% of the epoch runs vanilla SFT as warm-up, after which DIVE editing and vanilla optimization alternate in a repeating cycle of 1% DIVE bursts followed by 9% vanilla bursts. Evaluation is greedy decoding on the full test split of each dataset, and predictions are matched against the canonical label vocabulary via exact string match. All runs are executed on a Google Cloud TPU v5p-8.

We consider four text classification datasets that span sentence-pair semantic reasoning (*PAWS*), adversarially collected natural-language inference (*ANLI_r3*), and toxicity detection under both balanced and natural class distributions (*Civil_Comments_b*: 200K stratified 50/50; *Civil_Comments*: the full 1.8M natural distribution with $\sim$8% toxic). These four datasets differ substantially in scale (49K–1.8M training examples), number of classes (2–3), and difficulty, allowing us to probe DIVE's behavior across a range of LLM fine-tuning regimes.

Results and Analysis. Table 4 reports test loss and test accuracy of vanilla SFT and DIVE. *DIVE achieves lower test loss than vanilla SFT on every dataset without exception*—the reduction ranges from 2.2% (*PAWS*: 0.1584→0.1549) and 3.7% (*Civil_Comments*: 0.0702→0.0676) on the two saturated datasets to 8.4% (*ANLI_r3*: 0.8049→0.7374) and 15.8% (*Civil_Comments_b*: 0.1366→0.1150) on the two datasets that leave the model meaningful headroom. This uniform reduction in test loss shows that correcting the sign of detrimental samples yields better-calibrated predictions on held-out data across the full range of LLM fine-tuning regimes we consider.

On accuracy, DIVE delivers substantial gains on the two datasets where the vanilla baseline has meaningful headroom—*ANLI_r3* (+1.82 points, from 68.06 to 69.88) and *Civil_Comments_b* (+2.13 points, from 83.77 to 85.90)—while remaining essentially on par on *PAWS* and *Civil_Comments*, where vanilla SFT already reaches $\sim$95% and the room for improvement is small. The magnitude of the accuracy gain correlates with the noise level and task difficulty implicit in each dataset: *ANLI_r3* is adversarially curated so its training set contains many hard or ambiguous examples whose per-sample gradients frequently disagree with the batch consensus, and *Civil_Comments_b* contains inherent human-annotation noise (toxicity thresholded at 0.5 without adjudication). In contrast, *PAWS* and full *Civil_Comments* are cleaner or dominated by an easy majority class, leaving fewer detrimental samples for DIVE to correct. Crucially, the uniform improvement in test loss coupled with preserved accuracy on the saturated datasets shows that DIVE's editing never degrades the model in the easy regime: the sign flip is triggered only for samples whose gradient direction actually disagrees with the batch mean, so on well-fit data the intervention rate is naturally low. These findings confirm that DIVE is directly applicable to large-model text fine-tuning, with no algorithmic modification other than restricting influence computation to the trainable LoRA subspace.

6 Conclusion

In this paper, we proposed Dynamic Influence-based Valuation and Editing (DIVE), a lightweight and dynamic data curation framework designed to handle noisy and detrimental data during training. Unlike existing methods that primarily focus on identifying and removing detrimental samples, DIVE goes a step further by transforming detrimental samples into beneficial ones, effectively recovering informative underlying signals rather than discarding them. Through extensive experiments across standard image benchmarks and LLM fine-tuning regimes, DIVE consistently demonstrates superior performance and computational efficiency, remaining highly effective even in the absence of clean validation data via self-influence. Our findings highlight the importance of dynamic data valuation and open up promising avenues for repurposing, rather than discarding, noisy training data in large-scale model optimization.

AI USE STATEMENT

In this work, we used generative AI tools to assist with the implementation of our methods and coding of the data mismatch procedure that we designed. We also used AI tools for minor feedback on how to present related formulations.

We did not use generative AI to design the core methodology or experiments, propose hypotheses, develop theoretical models or conceptual frameworks, formulate mathematical claims or proofs, generate dataset content, interpret experimental results, or conduct literature searches or citation selection.

Additionally, we used generative AI tools for code debugging and optimization, experiment scripts, plotting and table formatting, and used large language models to polish the manuscript language and improve readability. We reviewed and tested all AI-assisted code in the actual experiments, and independently verified all reported results and final manuscript content. We take responsibility for the final manuscript and all reported results.

REPRODUCIBILITY STATEMENT

We release our code and implementation at `https://anonymous.4open.science/r/DIVE`. The repository contains reproducibility artifacts for all experiments. All experiments are implemented in JAX using TPU v5p-8 hardware. Details on environments and datasets, hyperparameters, and training and evaluation procedures are provided in Appendix, while launch commands are provided in the corresponding repository READMEs.

REFERENCES

Naman Agarwal, Brian Bullins, and Elad Hazan. Second-order stochastic optimization for machine learning in linear time. *Journal of Machine Learning Research*, 2017.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, 2009.

Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101–mining discriminative components with random forests. In *European Conference on Computer Vision*, pp. 446–461. Springer, 2014.

Guillaume Charpiat, Nicolas Girard, Loris Felardos, and Yuliya Tarabalka. Input similarity from the neural network perspective. *Advances in Neural Information Processing Systems*, 2019.

Anshuman Chhabra, Peizhao Li, Prasant Mohapatra, and Hongfu Liu. What data benefits my classifier? enhancing model performance and interpretability through influence-based data selection. In *International Conference on Learning Representations*, 2024.

R Dennis Cook and Sanford Weisberg. *Residuals and influence in regression*. New York: Chapman and Hall, 1982.

Talfan Evans, Nikhil Parthasarathy, Hamza Merzic, and Olivier Henaff. Data curation via joint example selection further accelerates multimodal learning. *Advances in Neural Information Processing Systems*, 2024.

Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, 2015.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016.

Amirata Ghorbani and James Zou. Data shapley: Equitable valuation of data for machine learning. In *International Conference on Machine Learning*, 2019.

Roger Grosse, Juhan Bae, Cem Anil, Nelson Elhage, Alex Tamkin, Amirhossein Tajdini, Benoit Steiner, Dustin Li, Esin Durmus, Ethan Perez, et al. Studying large language model generalization with influence functions. *arXiv preprint arXiv:2308.03296*, 2023.

Zayd Hammoudeh and Daniel Lowd. Training data influence analysis and estimation: A survey. *Machine Learning*, 113(5):2351–2403, 2024.

Frank R Hampel. The influence curve and its role in robust estimation. *Journal of the American Statistical Association*, 1974.

Satoshi Hara, Atsushi Nitanda, and Takanori Maehara. Data cleansing for models trained with sgd. *Advances in Neural Information Processing Systems*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2016.

Muyang He, Shuo Yang, Tiejun Huang, and Bo Zhao. Large-scale dataset pruning with dynamic uncertainty. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nezihe Merve Gurel, Bo Li, Ce Zhang, Costas Spanos, and Dawn Song. Efficient task specific data valuation for nearest neighbor algorithms. *In International Conference on Very Large Data Bases Endowment*, 2018.

Ruoxi Jia, David Dao, Boxin Wang, Frances Ann Hubis, Nick Hynes, Nezihe Merve Gürel, Bo Li, Ce Zhang, Dawn Song, and Costas J Spanos. Towards efficient data valuation based on the Shapley value. In *International Conference on Artificial Intelligence and Statistics*, 2019.

Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, 2021.

Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. Nlnl: Negative learning for noisy labels. In *IEEE/CVF International Conference on Computer Vision*, 2019.

Youngdong Kim, Juseung Yun, Hyounguk Shon, and Junmo Kim. Joint negative and positive learning for noisy labels. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International Conference on Machine Learning*, 2017.

Narine Kokhlikyan, Kamalika Chaudhuri, and Saeed Mahloujifar. Z0-inf: Zeroth order approximation for data influence. *arXiv preprint arXiv:2510.11832*, 2025.

M Kumar, Benjamin Packer, and Daphne Koller. Self-paced learning for latent variable models. *Advances in Neural Information Processing Systems*, 2010.

Yongchan Kwon and James Zou. Beta Shapley: a Unified and Noise-reduced Data Valuation Framework for Machine Learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Yongchan Kwon, Eric Wu, Kevin Wu, and James Zou. Datainf: Efficiently estimating data influence in lora-tuned llms and diffusion models. *arXiv preprint arXiv:2310.00902*, 2023.

R Douglas Martin and Victor J Yohai. Influence functionals for time series. *The Annals of Statistics*, 1986.

Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Höltgen, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, 2022.

Sung Min Park, Kristian Georgiev, Andrew Ilyas, Guillaume Leclerc, and Aleksander Madry. Trak: Attributing model behavior at scale. *arXiv preprint arXiv:2303.14186*, 2023.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 2020.

Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Zhaopan Xu, Daquan Zhou, Lei Shang, Baigui Sun, Xuansong Xie, et al. Infobatch: Lossless training speed up by unbiased dynamic data pruning. *arXiv preprint arXiv:2303.04947*, 2023.

Ravi S Raju, Kyle Daruwalla, and Mikko Lipasti. Accelerating deep learning with dynamic data pruning. *arXiv preprint arXiv:2111.12621*, 2021.

Corentin Salaün, Xingchang Huang, Iliyan Georgiev, Niloy J Mitra, and Gurprit Singh. Multiple importance sampling for stochastic gradient optimization. In *International Conference on Pattern Recognition Applications and Methods*, 2025a.

Corentin Salaün, Xingchang Huang, Iliyan Georgiev, Niloy J Mitra, and Gurprit Singh. Online importance sampling for stochastic gradient optimization. In *International Conference on Pattern Recognition Applications and Methods*, 2025b.

Andrea Schioppa, Polina Zablotskaia, David Vilar, and Artem Sokolov. Scaling up influence functions. In *AAAI Conference on Artificial Intelligence*, 2022.

Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *International Conference on Machine Learning*, 2019.

Petru Soviany, Radu Tudor Ionescu, Paolo Rota, and Nicu Sebe. Curriculum learning: A survey. *International Journal of Computer Vision*, 130(6):1526–1565, 2022.

Haoru Tan, Sitong Wu, Fei Du, Yukang Chen, Zhibin Wang, Fan Wang, and Xiaojuan Qi. Data pruning via moving-one-sample-out. *Advances in Neural Information Processing Systems*, 2023.

Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

Vishaal Udandarao, Nikhil Parthasarathy, Muhammad Ferjad Naeem, Talfan Evans, Samuel Albanie, Federico Tombari, Yongqin Xian, Alessio Tonioni, and Olivier J Hénaff. Active data curation effectively distills large-scale multimodal models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Conference*, 2025.

Jiachen T Wang, Prateek Mittal, Dawn Song, and Ruoxi Jia. Data shapley in one training run. *arXiv preprint arXiv:2406.11011*, 2024.

Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4555–4576, 2021.

Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. In *International Conference on Learning Representations*, 2022.

Ziao Yang, Longbo Huang, and Hongfu Liu. Layer-aware influence for online data valuation estimation. *arXiv preprint arXiv:2510.16007*, 2025.

Ziao Yang, Han Yue, Jian Chen, and Hongfu Liu. Revisit, extend, and enhance hessian-free influence functions. *Transactions on Machine Learning Research*, 2026.

APPENDIX

A DESCRIPTIONS OF DATASETS AND MODELS

Datasets. We evaluate DIVE on six noisy-label image classification benchmarks, summarized in Table 5. *CIFAR-10N* and *CIFAR-100N* (Wei et al., 2022) augment the original clean *CIFAR-10/CIFAR-100* training images with human re-annotations collected via Amazon Mechanical Turk. We use all three released label sets for CIFAR-10N — *aggregate*, *random*, and *worst*, denoted *CIFAR-10Na*, *CIFAR-10Nr*, and *CIFAR-10Nw* respectively — corresponding to noise rates of approximately 9%, 17%, and 40%, together with the single released label set for *CIFAR-100N* ($\sim$40% noise). Both datasets contain 50,000 training images over 10 and 100 classes respectively, evaluated on the standard 10,000-image *CIFAR* test split. *Animal-10N* (Song et al., 2019) consists of 50,000 training and 5,000 test images spanning 10 visually confusable animal classes (e.g., wolf/husky, hamster/guinea pig), human-annotation noise at approximately 8%. *Food-101N* (Bossard et al., 2014) contains 75,750 training and 25,250 test images across 101 food categories, with labels collected from web search results rather than human curation, carrying approximately 20% intrinsic label noise. Across the six benchmarks, noise rates range from $\sim$8% to $\sim$40%.

For *CIFAR-10N* and *CIFAR-100N*, which retain clean pre-noise labels for the training split, we construct the clean validation reference by holding out 1,000 clean-labeled samples from the training set before noise is applied, and resample this pool at each scoring step. *Animal-10N* and *Food-101N* do not provide clean labels for their training data, so we instead set aside a class-stratified 1/5 of each dataset's clean test split for validation and report test accuracy on the remaining 4/5.

Table 5: Summary statistics of the six image benchmarks used in our experiments.

Dataset	# Train	# Test	# Classes	Noise Type	Noise Rate
CIFAR-10Na	50,000	10,000	10	Human-annotated (aggregate)	$\sim$9%
CIFAR-10Nr	50,000	10,000	10	Human-annotated (random)	$\sim$17%
CIFAR-10Nw	50,000	10,000	10	Human-annotated (worst)	$\sim$40%
CIFAR-100N	50,000	10,000	100	Human-annotated	$\sim$40%
Animal-10N	50,000	5,000	10	Human-annotation	$\sim$8%
Food-101N	75,750	25,250	101	Intrinsic web-label	$\sim$20%

Models. All image experiments use a ResNet-9 (He et al., 2016) backbone, the compact fast-training architecture popularized by David Page's DAWNBench "How to Train Your ResNet" recipe. It consists of an initial 3×3 conv–batch-norm–ReLU stem (64 channels), followed by three down-sampling stages at 128, 256, and 512 channels (each a 3×3 conv–BN–ReLU block with 2×2 max-pooling), with two identity residual blocks — each two stacked conv–BN–ReLU layers with a skip connection — inserted after the 128-channel and 512-channel stages. A final *global* max-pool (rather than a fixed-window pool) collapses the spatial map to a single 512-dimensional feature vector per image regardless of its size, making the backbone resolution-adaptive: the same 512-d output applies uniformly across all six benchmarks despite their differing native resolutions (32×32 for *CIFAR-10N/100N*, 64×64 for *Animal-10N*, 96×96 for Food-101N). This backbone feeds into a three-layer MLP classification head ($512 \to 256 \to 128 \to$ #classes) with ReLU activations between layers. By default, sample-level influence scores are computed using gradients of only the *final* dense layer of the head, matching the paper's reported default configuration ("last-1").

B ADDITIONAL EXPERIMENTAL RESULTS

The in-depth study in the main text uses *CIFAR-10N-w*, whose labels are corrupted by synthetic noise. To confirm that the same behavior holds under real-world noise, we repeat the analysis on *Animal-10N* and *Food101-N*, shown in Figures 2 and 3. Both datasets tell the same story as *CIFAR-10N-w*. After the warm-up phase, DIVE attains the highest test accuracy among the curation methods, and it keeps correcting a sizeable, time-varying fraction of the training set rather than a fixed subset. Tracking individual samples reveals the mechanism behind this: the sign of a sample's influence is not stable,

so a sample that is beneficial at one stage of training can become detrimental later. The warm-up, editing-frequency, and scoring-layer ablations lead to the same conclusion as before, with DIVE reaching higher accuracy than GHOST and LAI at a lower training overhead. The main difference from *CIFAR-10N-w* is that these datasets carry less noise, so the influence scores gradually shrink toward zero as the model converges and most of the sign changes occur in the earlier epochs.

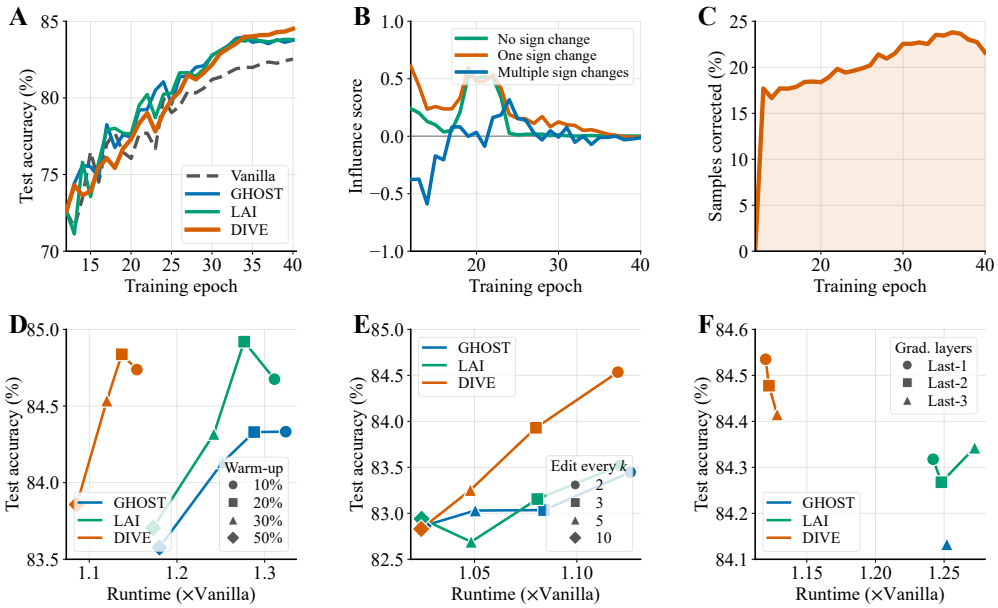

Figure 2: In-depth exploration of DIVE on *Animal-10N*.

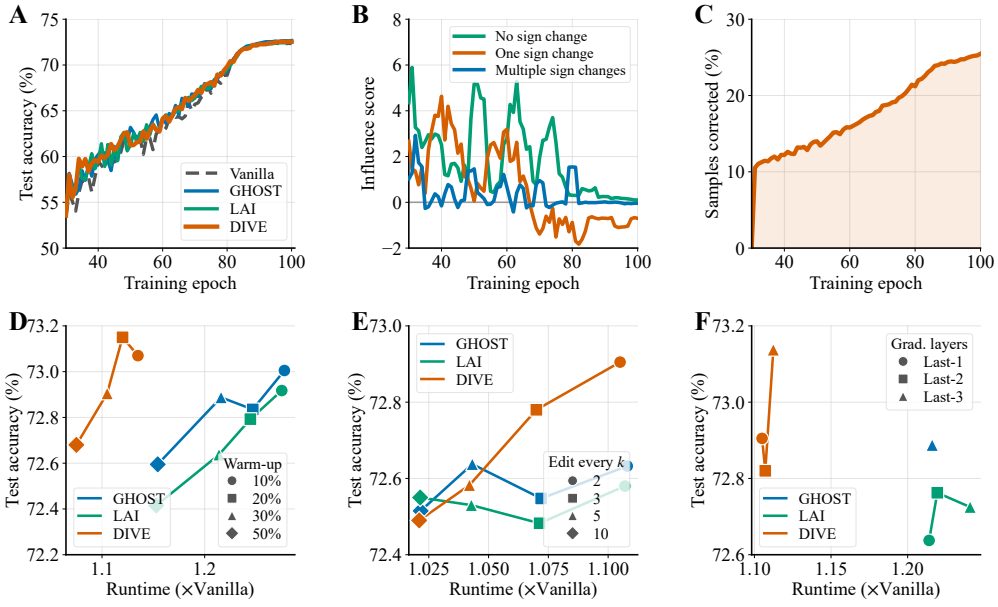

Figure 3: In-depth exploration of DIVE on *Food-101N*.