GradTrack: Detecting Noisy Labels via Temporal Trajectories of Class-wise Gradient Misalignment

Townim Faisal Chowdhury[1] **Ta Duc Huy**[1] **Vu Minh Hieu Phan**[1] **Anton van den Hengel**[1]
Johan W. Verjans[1] **Gustavo Carneiro**[3] **Zhibin Liao**[2,1]*

[1]Australian Institute for Machine Learning, Adelaide University, Australia
[2]School of Computer Science and Information Technology, Adelaide University, Australia
[3]University of Surrey, United Kingdom

zhibin.liao@adelaide.edu.au

Abstract

Noisy-label detection aims to identify incorrectly labeled training samples so that their adverse impact on model learning can be mitigated. Existing methods typically rely on loss, prediction confidence, or gradient magnitude signals, which primarily capture the scale of prediction errors. While effective, such scalar signals overlook the geometric structure and temporal dynamics of gradient evolution during training. In this work, we show that gradient magnitude provides a useful entry point for analyzing the gradient distortion induced by noisy-label samples, enabling us to quantify their deviation from the gradients that would be generated by clean labels. In this light, we propose **GradTrack**, a simple and effective framework for detecting noisy-label samples by estimating class-wise gradient directions likely to be induced by clean labels and measuring the alignment of training samples with these directions. In each training epoch, GradTrack partitions samples based on gradient magnitude and approximate clean-label gradient directions to compute a *class-wise gradient misalignment score*. These scores are used to rank training samples, which are then aggregated across training epochs to form a *temporal rank trajectory*. Finally, a Gaussian mixture model is applied to detect noisy-label samples. Extensive experiments on both synthetic and real-world noisy-label benchmarks demonstrate that GradTrack achieves strong detection performance, particularly under instance-dependent noise, and can further improve existing learning-with-noisy-labels pipelines as a plug-in sample selection module. Our code is publicly available at `https://github.com/AIML-MED/GradTrack`.

1 Introduction

Deep neural networks achieve impressive performance across visual recognition tasks, but they are notably vulnerable to label noise. Such noise is common in web-crawled and large-scale medical datasets [1, 38], where inaccurate annotations can substantially degrade generalization. This challenge has motivated extensive research on learning with noisy labels (LNL), whose goal is to identify corrupted samples and reduce their influence during training.

A common approach to noisy-label detection is to train a supervised noise detector. Such detectors are often learned by injecting synthetic symmetric or asymmetric noise into clean datasets, thereby creating known clean–noisy pairs. While convenient for controlled evaluation, this setting does not reflect real-world conditions, where mislabeling is often instance-dependent [50] and semantically

*Corresponding author

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

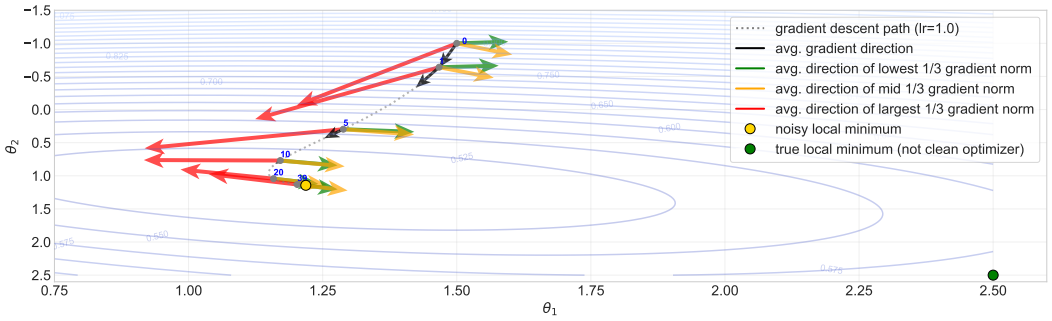


Figure 1: Gradient dynamics illustrated in the parameter space of a simple model working on a toy dataset with synthetic label noise, showing the model parameter update trajectory on the loss surface (the contours) toward the noisy local minimum in 40 epochs. The averaged lowest 1/3 and middle 1/3 gradient directions point more closely to the true local minimum. The largest 1/3 gradient direction drags the optimization toward a different direction, causing the average gradient over all samples to converge to the noisy local minimum.

correlated (*e.g.,* wolf *vs.* husky). As a result, synthetic corruption only weakly captures feature-dependent noise patterns and can lead to overly optimistic conclusions about robustness under real-world label noise.

Another line of work estimates label reliability from scalar training signals, such as loss [23], confidence [31], or related training statistics. These quantities are simple to compute and broadly useful for filtering samples, but they mainly describe the strength of the model's response at the output level. They do not reveal how a sample shapes optimization through its gradient, nor whether that influence is consistent with the dominant clean training dynamics.

In practice, two samples may have similar loss values or confidence scores while inducing parameter updates in very different directions. A scalar signal can indicate how incorrect the current prediction is, but it cannot distinguish whether the resulting gradient reinforces the dominant clean objective or conflicts with it. This distinction matters under label noise: difficult but correctly labeled samples may still support clean optimization, whereas mislabeled samples often introduce persistent directional disagreement. Figure 1 illustrates this intuition using a toy example. After sorting per-sample gradients by magnitude and partitioning them into three equal groups, we observe that the average gradient directions of the low- and medium-magnitude groups tend to guide the optimization toward the true local minimum. In contrast, the high-magnitude group pulls the optimization toward a noisy minimum, driven by its disproportionately large gradients. This observation suggests that gradient magnitude provides a useful coarse structure, but reliable noisy-label detection requires capturing both the underlying gradient geometry and its evolution over the course of training.

This motivates our proposed framework, **GradTrack**, which analyzes gradient alignment during optimization and aggregates these signals to produce a temporally robust noise-detection outcome. At each epoch, GradTrack partitions samples by gradient magnitude into low-, medium-, and high-magnitude groups. It then estimates class-wise clean gradient directions from the low- and medium-magnitude groups, refined using a denoising basis computed via Singular Value Decomposition (SVD). Each sample is assigned a *class-wise gradient misalignment score*, defined as its misalignment with the clean gradient direction of its labeled class within the same denoising basis; higher scores indicate a greater likelihood of being noisy. Rather than directly averaging these signals, GradTrack converts them into per-epoch noisiness rankings and aggregates them across training to form a *temporal rank trajectory*. We further apply SVD-based denoising to mitigate mirror rank inconsistencies across epochs, followed by a two-component Gaussian Mixture Model (GMM) to separate clean and noisy-label samples. This design enables GradTrack to distinguish persistent noisy behavior from transient training difficulty across both synthetic and real-world corruption settings.

To summarize, our main contributions are:

- We introduce **GradTrack**, a novel noisy-label detection framework that models label noise by analyzing gradient geometry with respect to an estimated clean-label direction.

- We develop a robust temporal aggregation scheme that ranks per-epoch noisiness estimates and employs a GMM to automatically separate clean and noisy samples without requiring manually specified thresholds using task knowledge (*e.g.,* noise rate).
- Evaluated on 11 synthetic and real-world noisy-label benchmarks, GradTrack achieves strong detection performance, particularly under instance-dependent noise, and can serve as a plug-in sample selection module for existing learning-with-noisy-labels pipelines.

2 Related Work

Learning with noisy labels (LNL) has been extensively studied [5]. The most relevant directions for our work are *sample selection* methods, which aim to identify corrupted samples, and *noise-robust learning* methods, which aim to reduce their impact during training.

Sample Selection Methods. A large body of work exploits the memorization effect of deep networks [3], using the observation that clean samples are often fitted earlier than noisy ones. Many methods therefore rely on the *small-loss principle* [11, 14, 23, 19], while Co-Teaching and its variants [11, 55, 49] use two networks to mitigate confirmation bias during sample selection. DivideMix [23] models the loss distribution with a Gaussian mixture model, and Confident Learning [31] estimates label errors from prediction confidences. Representation-based approaches [61, 48, 21] instead use latent features to separate clean and noisy samples, with SIMIFEAT [61] being a representative example. Closer to our work, gradient- and training-dynamics methods analyze how samples influence optimization over time. TracIn [36] uses self-influence derived from gradient updates, while AUM [35], L2D [16], Kim *etal.* [20], HMW [56], and CT [32] model training dynamics rather than relying on single-epoch statistics. However, existing methods typically use scalar loss, confidence, or influence signals, and do not explicitly model class-conditioned gradient geometry together with its temporal evolution. Our method addresses this gap by analyzing per-sample gradient conflict through both magnitude and geometric disagreement over training.

Noise-Robust Learning. Another major line of research focuses on learning objectives that are robust to label corruption. This includes robust loss functions [57, 43, 10, 58, 44, 29], regularization-based methods [27, 8, 28, 15, 60, 24], loss correction via transition modeling [34, 51, 4, 53, 18, 7], and label-correction approaches [41, 54, 2]. More recent methods combine robust training with explicit sample filtering, re-weighting, or relabeling [17, 39, 40, 12, 59], often within semi-supervised frameworks such as DivideMix [23, 42]. These methods are complementary to noisy-label detection: robust training reduces the effect of corrupted labels, while detection methods provide sample-level reliability estimates. In this paper, we focus on the detection side and additionally show that GradTrack can be used as a plug-in sample-selection module for existing LNL pipelines.

3 Preliminaries and Motivation

3.1 Problem Formulation

We study the setting of multi-class classification, where inputs belong to an image space $\mathcal{X}$ and labels are chosen from $\mathcal{Y} = \{1, \dots, C\}$, here C denotes the largest class index. Ideally, one would train with a clean dataset $D = \{(\mathbf{x}_i, y_i)\}_{i=1}^N$, sampled from the true joint distribution. Yet in real-world applications, the clean labels are hidden, and we only have access to the potentially corrupted labels. Beside D, we note a noisy collection $\widetilde{D} = \{(\mathbf{x}_i, \widetilde{y}_i), k_i\}_{i=1}^N$, where an individual label $\widetilde{y}_i$ may differ from the unobservable ground truth y_i. The variable k_i is also unobservable during the training but accessible during the test phase for evaluation, and it denotes whether the label is clean or noisy, *i.e.,* $k_i = \mathbb{1}(y_i \neq \widetilde{y}_i)$. The noise label detection problem is therefore identifying the collection of samples with $k_i = 1$.

Consider a model that produces classification logits $\mathbf{z} = F(\mathbf{x} \mid \theta)$ with parameters θ for an input $\mathbf{x}$. We distinguish the model into two components: (1) a *feature encoder* that produces a latent feature representation $\mathbf{h} = F_{\text{feat}}(\mathbf{x} \mid \theta_{\text{feat}})$, and (2) a *classification head* that produces $\mathbf{z} = F_{\text{cls}}(\mathbf{h} \mid \mathbf{W})$. Given a loss function $\mathcal{L}(\mathbf{z}, \widetilde{y})$ (Cross Entropy in our case), we record the gradient of each sample with respect to (*w.r.t.*) the classification head's parameters $\mathbf{W}$, written as $\nabla_{\mathbf{W}} \mathcal{L}(\mathbf{z}, \widetilde{y})$, across training iterations. We keep the encoder parameters θ_{feat} fixed at their pretrained values. Although it is possible to train the encoder and track its gradients, the encoder parameter space is substantially larger. Moreover,

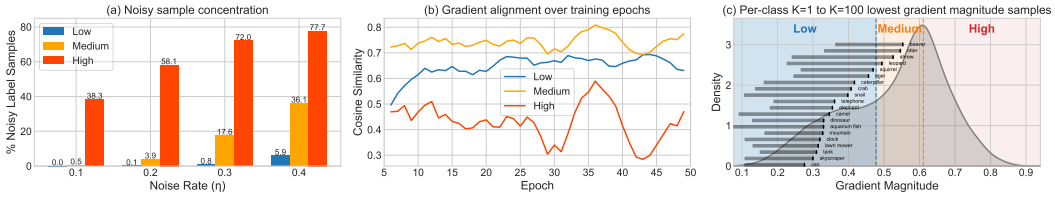


Figure 2: **Gradient dynamics under label noise** on CIFAR-100 with instance-dependent noise using a frozen CLIP ViT-B/32 encoder. **(a)** Fraction of noisy samples in the low, medium, and high gradient-magnitude groups measured at noise rates. **(b)** Cosine similarity between the average gradient of each group and the *oracle clean-label gradient direction* over all training epochs at $\eta = 0.4$. The medium-magnitude group remains most aligned with the clean-label gradient. **(c)** Gradient-magnitude distribution. Each horizontal bar represents the bottom $K = 100$ gradient magnitudes for one of 20 randomly selected classes, ordered by the 100th (largest) value within each set. Shaded regions indicate low-, medium-, and high-magnitude groups of all classes. The results show that class-wise bottom-K gradients do not necessarily remain confined to the low-magnitude group.

gradients with respect to earlier layers tend to be less discriminative, as backpropagation scales them by shared model parameters through the chain rule, thereby attenuating sample-specific differences.

We compute each sample's gradient from saved checkpoints following the practice in TracIn [36], obtaining $\nabla^e_{\mathbf{W}} \mathcal{L}(\mathbf{z}, \widetilde{y})$, where e denotes the epoch index. Finally, we introduce a simplified notation $\mathbf{g}^e_i = \nabla^e_{\mathbf{W}} \mathcal{L}(\mathbf{z}_i, \widetilde{y}_i) \in \mathbb{R}^{(C \times d)}$ to denote the gradient of individual training samples indexed by i recorded at epoch e, where d denotes the length of $\mathbf{h}$. Aggregating these gradients across epochs yields the gradient dynamics of the sample:

$$G_i = \left[\mathbf{g}^1_i, \mathbf{g}^2_i, \ldots, \mathbf{g}^E_i\right] \in \mathbb{R}^{(C \times d) \times E}, \tag{1}$$

where E is the total number of epochs.

3.2 Gradient Dynamics under Label Noise

To motivate our method, we study how gradient coherence changes under label noise. Prior work on coherent gradients [6] hypothesized that similar training examples induce similar gradients, so their contributions reinforce one another and produce stronger shared directions during optimization. We ask how this behavior changes when some labels are corrupted. As shown in Figure 2, three empirical patterns emerge. At each epoch e, we compute the gradient magnitudes $\|\mathbf{g}^e_i\|_2$ for all samples, sort them in ascending order, and partition them into three equal-sized groups.

Observation 1: Gradient magnitudes stratify clean and noisy label samples. Figure 2-(a) shows the fraction of noisy-label samples in each group across noise rates $\eta \in \{0.1, 0.2, 0.3, 0.4\}$, measured with access to ground-truth labels. Intuitively, the low-magnitude group contains the fewest corrupted samples, whereas the high-magnitude group contains the most. This pattern is consistently observed across all 11 synthetic and real-world noisy-label benchmark settings; supporting statistics for Observation 1 are reported in Appendix B.

Observation 2: Low- and medium-magnitude groups provide useful clean-gradient approximations. We refer to the average gradient computed from training samples with their true labels (y_i) as the *oracle clean-label gradient direction*. Figure 2-(b) shows that the medium group has the highest cosine similarity with this reference in the illustrated setting. Although the low-magnitude group is cleaner, we believe its well-fitted samples contribute weak gradients, whereas harder clean samples in the medium-magnitude group provide stronger gradients that exert greater influence on the optimization trajectory. Accordingly, the medium-magnitude group exhibits the highest alignment with the oracle clean-label gradient direction in 8 of the 11 settings. The exceptions are the three settings with symmetric noise at $\eta = 0.6$, where heavy contamination by noisy-label samples compromises its approximation, making the low-magnitude group a more reliable estimate. Together, these findings motivate incorporating both low- and medium-magnitude samples when approximating the clean-label gradient direction, rather than relying exclusively on either group. In addition, Figure 2-(c) shows that the lowest-magnitude gradients of some classes extend beyond the global low-magnitude

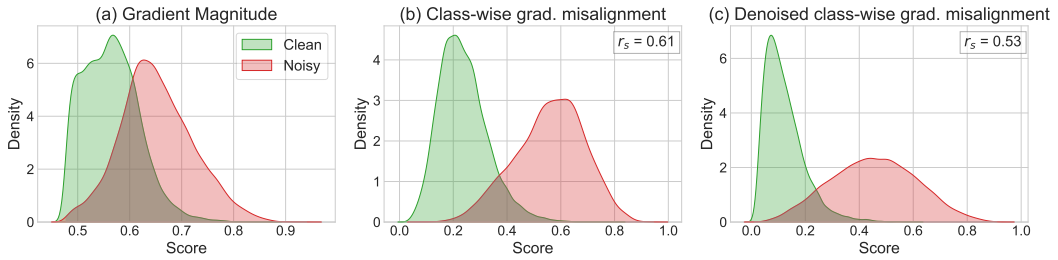


Figure 3: **(a)** Normalized gradient magnitude distributions for clean- and noisy-label samples, with per-sample scores averaged over post-warmup training epochs. **(b)** Class-wise gradient misalignment score distributions (Eq. 3). **(c)** Class-wise gradient misalignment score distributions after projection onto the denoised gradient basis (Eq. 6). In (b) and (c), r_s denotes Spearman's rank correlation coefficient between per-sample gradient magnitude and the corresponding misalignment scores, computed separately at each epoch and averaged over post-warmup training epochs.

group, highlighting the need to include the medium-magnitude group for the class-wise gradient references. Supporting statistics and exceptions to Observation 2 are also reported in Appendix B.

Observation 3: Gradient misalignment provides complementary signals to gradient magnitude for detecting noisy labels. Figure 3 compares gradient magnitude with class-wise and denoised class-wise gradient misalignment scores, illustrating their average score distributions across clean- and noisy-label groups. The distributions are computed over post-warmup training epochs for the medium- and high-magnitude groups on CIFAR-100 with instance-dependent noise at $\eta = 0.4$. Although class-wise gradient misalignment relies on gradient magnitude to initialize its estimation of the clean-label gradient, the resulting scores exhibit different characteristics in separating clean- and noisy-label samples. The Spearman's rank correlation coefficients of $r_s = 0.61$ and 0.53 reported in (b) and (c), respectively, are computed against gradient magnitude at each epoch and averaged over post-warmup epochs. These moderate correlations suggest that gradient misalignment may provide complementary information for distinguishing clean- and noisy-label samples, motivating the combination of both signals in Sec. 4.4. In terms of noisy-label detection performance, class-wise misalignment outperforms gradient magnitude in 7 of the 11 settings, whereas gradient magnitude performs better in the remaining four (see Appendix B).

4 Methodology

Motivated by the observations, we adopt the same three group partition and combine the low- and medium-magnitude groups into a predominantly clean sample group, denoted by $\mathcal{S}^e_{\text{l+m}} = \{1, \ldots, \frac{2}{3}N\}$ representing the corresponding sample indices after sorting by their gradient magnitudes. This reduces the influence of noisy-label samples in subsequent gradient alignment analysis.

4.1 Class-wise Clean-label Gradient Approximation

Figure 2-(b) and (c) suggest that the oracle clean-label gradient direction may be dominated by a subset of classes with higher classification difficulty. Consequently, a global clean-label gradient estimated across all classes can be biased, leading the alignment measure to preferentially treat samples from easier classes as cleaner. To mitigate this effect, we compute class-wise clean-label gradient directions and compare each training sample against the direction corresponding to its label ($\widetilde{y}_i$). Specifically, for class c, we select the bottom-K samples labeled as c from $\mathcal{S}^e_{\text{l+m}}$ to compute an *approximate class-wise clean-label gradient direction*:

$$\boldsymbol{\mu}^e_c = \frac{1}{|\mathcal{S}^e_c|} \sum_{j \in \mathcal{S}^e_c} \mathbf{g}^e_j, \tag{2}$$

where $\mathcal{S}^e_c = \text{bottom}_K\left(\{\mathbf{g}^e_i \mid \widetilde{y}_i = c\}_{i \in \mathcal{S}^e_{\text{l+m}}}\right)$, and $\text{bottom}_K(\cdot)$ returns the indices of the K samples with the smallest gradient magnitudes.

Comparing with $\boldsymbol{\mu}_c^e$ yields an initial estimate of noise likelihood based on gradient alignment. The proposed *class-wise gradient misalignment score* is defined as a cosine similarity measure:

$$m_i^e = 1 - \cos(\boldsymbol{\mu}_c^e, \mathbf{g}_i^e)\,, \tag{3}$$

for *all* training samples with $\widetilde{y}_i = c$. Given that the approximate class-wise clean-label gradient directions are most likely still influenced by noisy-label samples, we introduce a denoising step prior to computing the misalignment score.

4.2 Denoising by SVD

Observation 1 suggests that noisy samples are rare in the low-magnitude group but become more prevalent in the medium-magnitude group. We therefore use $\mathcal{S}_{\text{l+m}}^e$ in this step to ensure that clean samples (including both easy and hard clean samples) remain dominant, while noisy-label samples are treated as targets for denoising.

We first compute the unnormalized scatter matrix of gradients over $\mathcal{S}_{\text{l+m}}^e$ as:

$$\mathbf{C}_{\text{l+m}}^e = \sum_{j \in \mathcal{S}_{\text{l+m}}^e} \mathbf{g}_j^e \mathbf{g}_j^{e\top}, \tag{4}$$

which captures the aggregate second-order structure of the selected gradients. We then apply SVD[2] to extract its dominant directions. We retain the top-M right singular vectors ($\mathbf{v}$) whose cumulative explained variance exceeds a threshold λ, and form an orthonormal basis matrix:

$$\mathbf{V}^e = [\mathbf{v}_1^e, \ldots, \mathbf{v}_M^e] \in \mathbb{R}^{(C\times d)\times M}. \tag{5}$$

Since $\mathbf{C}_{\text{l+m}}^e$ is unnormalized, the resulting orthonormal basis is expected to capture the dominant gradient directions supported by the majority of clean samples, while isolated high-magnitude noisy gradients are less likely to influence the basis. The resulting projector $\mathbf{P}^e = \mathbf{V}^e \mathbf{V}^{e\top}$ is then used in the next step to denoise for the misalignment score.

4.3 Denoised Class-wise Gradient Misalignment Score

We now incorporate Eq. (3) by applying the projector $\mathbf{P}$ so that the updated class-wise gradient misalignment score is:

$$m_i^{e,\text{den}} = 1 - \cos(\mathbf{P}\boldsymbol{\mu}_c^e, \mathbf{P}\mathbf{g}_i^e)\,. \tag{6}$$

In fact, since cosine similarity after projection onto an orthonormal basis can be computed equivalently using the corresponding low-dimensional representations, $\mathbf{P}$ in Eq. (6) can be replaced by $\mathbf{V}^{e\top}$ to reduce computational and memory overhead.

4.4 Temporal Rank Trajectory Modeling

The gradient magnitude $\|\mathbf{g}_i^e\|_2$ and denoised class-wise misalignment score $m_i^{e,\text{den}}$ provide different signals for noisy-label detection at each epoch. However, these signals are derived from model parameters at a single training snapshot and may therefore be unstable. To obtain a more robust estimate, we model their temporal consistency across training.

Given that the two signals follow different distributions, directly averaging them may bias the result toward one of the scores. To balance their contributions, we first convert each signal into a ranking. Let $r_i^{e,\text{mag}}, r_i^{e,\text{mis}} \in [0,1]$ denote the normalized ranks of sample i at epoch e, computed from $\|\mathbf{g}_i^e\|_2$ and $m_i^{e,\text{den}}$, respectively, across the training set. Each sample is then represented by its temporal rank trajectory:

$$\mathbf{r}_i = \left[r_i^{E_{\text{warm}}+1,\text{mag}}, \ldots, r_i^{E,\text{mag}}, r_i^{E_{\text{warm}}+1,\text{mis}}, \ldots, r_i^{E,\text{mis}}\right]^\top, \tag{7}$$

where $E_{\text{warm}} + 1$ denotes the first epoch after the warm-up phase, and we exclude the warm-up stage because gradients in this stage often share similar geometric characteristics, which provide limited discriminability to detect label noise.

[2]Note that, in principle, eigendecomposition can also be applied since $\mathbf{C}_{\text{l+m}}^e$ is positive semi-definite.

We then apply the same SVD-based denoising technique to the normalized trajectory vectors, retaining the top-T components whose cumulative explained variance exceeds a threshold ρ. This yields a denoised low-dimensional embedding $\mathbf{r}'_i \in \mathbb{R}^T$, which mitigates temporal rank inconsistencies.

Finally, we fit a two-component Gaussian Mixture Model (GMM) with full covariance to the embeddings $\{\mathbf{r}'_i\}_{i=1}^N$. As the mixture components are unlabeled, we identify the component with the higher average rank as corresponding to noisy-label samples. The GMM produced posterior probability:

$$p_i = \Pr(\text{noisy} \mid \mathbf{r}'_i) \tag{8}$$

serves as the final detection score, and we conclude $k_i = 1$, *i.e.,* $\widetilde{y}_i$ is noisy, if $p_i \geq 0.5$.

4.5 Integration with LNL Methods

As its name implies, the goal of learning-with-noisy-labels (LNL) methods is to maintain high classification accuracy even when training data contain corrupted annotations. GradTrack is architecture-agnostic and can be seamlessly integrated with a wide range of LNL frameworks.

Initialization. We first train a simple classifier F and run GradTrack, which acts like a noisy-label detector F_{det} to yield an initial partition of $\widetilde{D}$ into clean and noisy subsets. These noisy-label predictions are then fused with those from a target LNL method by taking the union of both:

$$k_{i,0} = k_{i,0}^{(\text{lnl})} \vee k_{i,0}^{(\text{det})}, \tag{9}$$

where $k_{i,0}^{(\text{lnl})}$ and $k_{i,0}^{(\text{det})}$ denote the initial (second subscript $t = 0$) noisy label assignments from the base LNL method and our GradTrack-based detector, respectively.

Iterative updates. To adapt to a common iterative training paradigm of the LNL methods (*i.e.,* iteratively filtering clean samples for training), we implement:

$$k_{i,t} = k_{i,t}^{(\text{lnl})} \wedge k_{i,t}^{(\text{det})}, \tag{10}$$

where we treat the LNL-trained classifier as F to support a faster estimation of noisy labels on the fly.

5 Experiments

5.1 Experimental Setup

Datasets. We evaluate on both synthetic and real-world noisy-label benchmarks. For *synthetic noise*, we use CIFAR-10, CIFAR-100 [22], and Tiny-ImageNet [47]. Following standard protocols [11, 50, 23], we consider symmetric, asymmetric, and instance-dependent noise. Symmetric noise flips labels uniformly at random; asymmetric noise uses semantically plausible class transitions, *e.g.,* Cat $\leftrightarrow$ Dog in CIFAR-10 and circular transitions within super-classes in CIFAR-100; instance-dependent noise [50] depends on both the clean class and sample features. Unless otherwise stated, we use symmetric noise with $\eta = 0.6$ (*Sym.* 0.6), asymmetric noise with $\eta = 0.3$ (*Asym.* 0.3), and instance-dependent noise with $\eta = 0.4$ (*Inst.* 0.4).

For *real-world noise*, we use CIFAR-10N and CIFAR-100N [45], which provide human-annotated noisy labels. On CIFAR-10N, we evaluate on the *Agg.* subset ($\eta \approx 0.09$) and the *Worst* subset ($\eta \approx 0.4$); on CIFAR-100N, we use the *Human* labels ($\eta \approx 0.4$). We also evaluate on WebVision [25], using the mini WebVision split (the first 50 Google classes) following [23], and on Clothing1M [52]. Together, these datasets cover synthetic and real noise, low- and high-noise regimes, and both small-scale and large-scale settings.

Competing Methods. We compare against state-of-the-art sample-selection methods, including AUM [35], CL [31], CORES [9], TracIn [36], SIMIFEAT [61], L2D [16], and CT [32]. Among them, L2D and CT are the most closely related, as they also model training dynamics through loss and confidence signals, respectively. For TracIn, we follow the implementation details in [61] for fair comparison. We further integrate GradTrack with recent LNL methods, including ELR [28], DivideMix [23], DISC [26], L2D [16], and DeFT [46], following Sec. 4.5; the resulting classification results are reported in Sec. 5.3.

Evaluation Metrics. For noisy-label detection, we report F1, following [61]. For WebVision and Clothing1M, where ground-truth corruption labels are unknown, we report downstream test accuracy.

Implementation Details. We use a frozen CLIP [37] ViT-B/32 image encoder with an MLP classification head of two hidden layers, and compute per-sample gradients with respect to the final linear classification layer only, consistent with Section 3.1. We train the head for 50 epochs on all datasets except CIFAR-10, where we train for 30 epochs. Optimization uses vanilla SGD without momentum, a fixed learning rate of 0.1, and no learning-rate schedule. When forming the temporal rank trajectories in Eq. (7), we discard the first 5 epochs as warm-up, *i.e.,* $E_{\text{warm}} = 5$. Unless otherwise stated, we set $K = 100$, $\rho = 0.85$, and $\lambda = 0.95$. All experiments are implemented in PyTorch [33] and run on a single Nvidia RTX A6000 GPU with 48GB memory.

Table 1: F1 scores (%) of noisy-label detection methods across 11 benchmark settings including synthetic and real-world noise using frozen CLIP image features. The *Avg.* column reports the arithmetic mean over all settings. Best and second-best results in each column are highlighted in **bold** and underlined, respectively.

Method	**CIFAR-10**			**CIFAR-10N**		**CIFAR-100**			**CIFAR-100N**	**Tiny-ImageNet**		**Avg.**
Noise type	Sym.	Asym.	Inst.	Worst	Agg.	Sym.	Asym.	Inst.	Human	Sym.	Inst.	
Noise rate (η)	0.6	0.3	0.4	0.4	0.09	0.6	0.3	0.4	0.4	0.6	0.4	
AUM [35]	95.51	73.67	79.46	87.01	76.34	91.36	61.67	77.94	66.94	**93.90**	81.29	80.74
CL [31]	95.82	85.57	72.46	77.97	68.60	86.84	65.59	68.72	69.77	88.50	75.20	77.73
CORES [9]	**97.32**	8.53	54.05	80.34	**80.50**	88.12	14.46	61.11	37.26	85.20	70.50	61.58
TracIn [36]	95.07	79.23	76.86	79.72	75.84	87.39	61.66	72.36	77.14	86.26	70.72	78.39
SIMIFEAT-V [61]	95.61	88.40	87.11	87.82	78.77	87.58	74.41	80.41	76.85	81.12	80.88	83.54
SIMIFEAT-R [61]	96.09	**88.72**	91.02	88.29	79.16	89.09	68.10	77.32	78.21	81.19	75.97	83.02
L2D [16]	95.78	65.60	81.00	87.71	59.68	90.19	53.31	71.58	72.18	91.46	72.87	76.49
CT [32]	95.63	76.87	79.61	86.99	67.77	90.78	63.62	70.88	76.17	87.50	74.20	79.09
GradTrack	97.29	87.48	**94.80**	**91.63**	69.23	**93.85**	**78.65**	**89.25**	**79.12**	93.75	**86.32**	**87.40**

5.2 Noisy-Label Detection Results

Table 1 reports F1 scores across 11 noisy-label detection settings. GradTrack achieves the highest average performance (87.40%), outperforming the strongest baseline, SIMIFEAT-V (83.54%). GradTrack attains the best results in 7 of the 11 settings, including all CIFAR-100 and CIFAR-100N configurations, as well as Tiny-ImageNet with instance-dependent noise. This indicates that the proposed class-wise gradient misalignment score and rank-trajectory-based aggregation are effective in solving challenging noisy-label tasks. Additional Area Under the Receiver Operating Characteristic Curve (AUC) results are provided in Appendix Table 8 for GradTrack (last column).

On CIFAR-10N Agg., GradTrack retains strong ranking performance (AUC 97.78%) but over-cleans: it flags approximately 15.15% of samples as noisy against a true noise rate of 9.01%, yielding 92.81% recall and 55.21% precision. In other words, a GMM decision threshold of 0.5 appears overly aggressive in low-noise settings, although manual threshold tuning should ideally be avoided. Appendix A.2 further examines this over-cleansing effect and shows that its effect is also related to and possibly counteracted by the cumulative variance threshold ρ. Increasing ρ from the default 0.85 to 0.90 improves F1 to 80.79%.

5.3 Learning with Noisy Labels Results

We integrate our approach with several LNL baselines, including Cross Entropy (CE), ELR [28], DivideMix [23], DISC [26], L2D [16], and DeFT [46]. As shown in Table 2, incorporating GradTrack consistently improves test accuracy across all three datasets. Notably, strong baselines such as DISC and DeFT also benefit from using the GradTrack noise detection module. These results demonstrate that GradTrack is complementary to existing LNL training pipelines and can enhance performance in challenging real-world noisy-label scenarios.

5.4 Ablation Studies

Component Ablation. We study the contribution of each component in the GradTrack design. (1) **Magnitude-Only** uses only gradient magnitude in the temporal rank trajectory in Eq. (7). (2) **Baseline** directly applies the class-wise gradient misalignment score m_i^e in Eq. (3) without the SVD-based denoising step described in Sec. 4.2. (3) **+ All gradients** adds the denoising step, but computes the scatter matrix using all samples, *i.e.,* $\mathcal{S}_{\text{l+m+h}}$ in Eq. (4). (4) **+** $\mathcal{S}_{\text{l+m}}$ removes the high-magnitude group

Table 2: Test accuracy (%) under real-world noise using the CLIP encoder.

Method	CIFAR-100N	WebVision	Clothing1M
CE	66.95	71.08	65.41
+Ours	**+2.90**	**+4.16**	**+3.70**
ELR [28]	72.18	78.48	71.44
+Ours	**+0.50**	**+1.39**	**+0.29**
DivideMix [23]	70.12	79.19	68.60
+Ours	**+1.35**	**+1.13**	**+0.31**
DISC [26]	71.05	80.08	70.57
+Ours	**+0.50**	**+0.78**	**+0.54**
L2D [16]	71.70	80.23	70.62
+Ours	**+0.25**	**+0.78**	**+0.63**
DeFT [46]	69.33	77.20	71.60
+Ours	**+2.50**	**+1.52**	**+1.25**

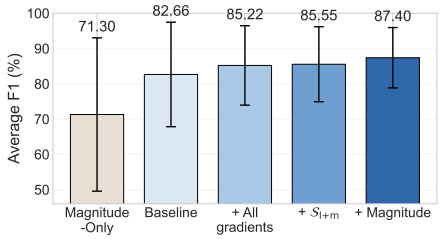


Figure 4: Ablation of GradTrack components measured by average F1 across 11 settings. Error bars denote the standard deviation. Results for individual settings are reported in Appendix Table 8.

from (3), yielding the proposed formulation of Eq. (6). Finally, (5) **+ Magnitude** combines (4) and (1), resulting in the proposed temporal rank trajectory in Eq. (7).

Figure 4 reports the average F1 score across all 11 benchmark settings. The largest improvement arises from introducing the class-wise gradient misalignment score, indicating that directional disagreement with respect to the labeled class provides a strong signal for noisy-label detection. Applying SVD-based denoising further improves the average F1. Excluding high-magnitude gradients yields marginal additional gains but reduces the standard deviation, indicating more stable performance. Finally, jointly leveraging misalignment and magnitude signals over the training trajectory leads to further improvements, indicating that these two signals are complementary.

Additional analyzes. Additional analyzes are provided in the Appendix, including sensitivity to λ (App. A.1) and ρ (App. A.2), gradient magnitude-based partitioning (App. A.3), temporal rank trajectory aggregation (App. A.4), the size of K for estimating class-wise clean gradient direction (App. A.5), alternative clustering algorithms (App. A.6), and additional validation using ResNet-50 [13] feature encoder (App. C).

5.5 Robustness to Noise Severity

Figure 5 evaluates noisy-label detection performance under increasing levels of instance-dependent noise on CIFAR-10, CIFAR-100 and Tiny-ImageNet. We compare GradTrack with SIMIFEAT-V, the strongest state-of-the-art baseline. GradTrack consistently outperforms SIMIFEAT-V across the sampled noise rates, and remains competitive at low noise levels on CIFAR-10, indicating stronger robustness to varying noise severity.

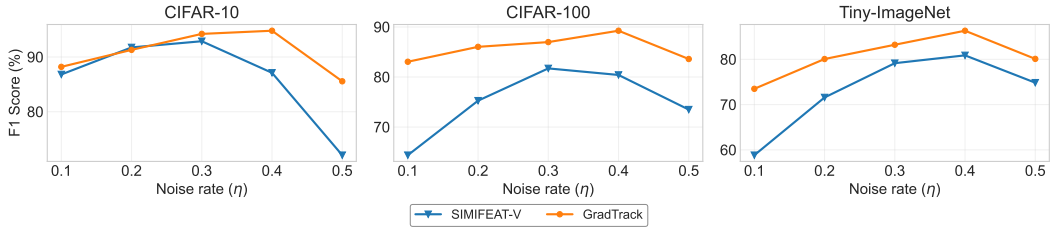


Figure 5: Robustness to noise severity measured by F1 under instance-dependent noise with varying noise rates on CIFAR-10, CIFAR-100 and Tiny-ImageNet.

6 Conclusion

In this paper, we introduced GradTrack, a noisy-label detection framework that captures temporal rank trajectories of gradient magnitude and class-wise gradient alignment to identify samples that persistently conflict with the clean-label optimization geometry. Experiments on both synthetic

and real-world benchmarks demonstrate strong detection performance, particularly under instance-dependent noise, and highlight its effectiveness as a plug-in sample selection module for existing LNL pipelines.

Limitations. (1) GradTrack leverages frozen feature encoders and computes gradients with respect to the classification head, providing a practical design that minimises computational overhead. While using gradients from the full parameter space may capture additional signals, it can also lead to more diffuse gradients due to the chain rule, potentially making gradient directions less distinguishable from one another. Nevertheless, examining gradients of the full model remains an interesting direction and will be explored in future work. (2) From a practical standpoint, GradTrack is more computationally expensive than lightweight scalar-based detectors, as it examines temporal gradient geometry evolution. Profiling on Tiny-ImageNet shows that per-sample gradient extraction accounts for 98.97% of the measured runtime, while subsequent GradTrack-imposed processing accounts for 1.03%; the runtime breakdown is provided in Appendix D. Although restricting gradient computation to the classification head reduces computational overhead, GradTrack is better suited for offline noisy-label detection or periodic filtering of noisy-label samples during training.

Broader Impact. GradTrack represents a step forward in noisy-label learning by providing additional noise-revealing signals through the association between training gradient geometry and its clean-label counterpart. More importantly, we demonstrate a pathway to capture and quantify gradient behavior of training samples, an aspect that is difficult to analyze yet reflects how the model attempts to *deal with* inputs during optimization. The impact of gradient behavior analysis extends beyond noisy-label detection and may inspire approaches in anomaly detection and out-of-distribution detection. More broadly, it offers insights toward a deeper understanding and improved robustness in learning systems.

References

[1] Paul Albert, Diego Ortego, Eric Arazo, Noel E O'Connor, and Kevin McGuinness. Addressing out-of-distribution label noise in webly-labelled data. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 392–401, 2022.

[2] Eric Arazo, Diego Ortego, Paul Albert, Noel O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *International conference on machine learning*, pages 312–321. PMLR, 2019.

[3] Devansh Arpit, Stanisław Jastrzbski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pages 233–242. PMLR, 2017.

[4] Antonin Berthon, Bo Han, Gang Niu, Tongliang Liu, and Masashi Sugiyama. Confidence scores make instance-dependent label-noise learning possible. In *International conference on machine learning*, pages 825–836. PMLR, 2021.

[5] Gustavo Carneiro. *Machine Learning with Noisy Labels: Definitions, Theory, Techniques and Solutions*. Elsevier, 2024.

[6] Satrajit Chatterjee. Coherent gradients: An approach to understanding generalization in gradient descent-based optimization. In *International Conference on Learning Representations*, 2020.

[7] De Cheng, Tongliang Liu, Yixiong Ning, Nannan Wang, Bo Han, Gang Niu, Xinbo Gao, and Masashi Sugiyama. Instance-dependent label-noise learning with manifold-regularized transition matrix estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16630–16639, 2022.

[8] De Cheng, Yixiong Ning, Nannan Wang, Xinbo Gao, Heng Yang, Yuxuan Du, Bo Han, and Tongliang Liu. Class-dependent label-noise learning with cycle-consistency regularization. *Advances in Neural Information Processing Systems*, 35:11104–11116, 2022.

[9] Hao Cheng, Zhaowei Zhu, Xingyu Li, Yifei Gong, Xing Sun, and Yang Liu. Learning with instance-dependent label noise: A sample sieve approach. *arXiv preprint arXiv:2010.02347*, 2020.

[10] Aritra Ghosh, Himanshu Kumar, and P Shanti Sastry. Robust loss functions under label noise for deep neural networks. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

[11] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

[12] Jiangfan Han, Ping Luo, and Xiaogang Wang. Deep self-learning from noisy labels. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5138–5147, 2019.

[13] 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*, pages 770–778, 2016.

[14] Jinchi Huang, Lie Qu, Rongfei Jia, and Binqiang Zhao. O2u-net: A simple noisy label detection approach for deep neural networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3326–3334, 2019.

[15] Ahmet Iscen, Jack Valmadre, Anurag Arnab, and Cordelia Schmid. Learning with neighbor consistency for noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4672–4681, 2022.

[16] Qingrui Jia, Xuhong Li, Lei Yu, Jiang Bian, Penghao Zhao, Shupeng Li, Haoyi Xiong, and Dejing Dou. Learning from training dynamics: Identifying mislabeled data beyond manually designed features. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 8041–8049, 2023.

[17] 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*, pages 2304–2313. PMLR, 2018.

[18] Zhimeng Jiang, Kaixiong Zhou, Zirui Liu, Li Li, Rui Chen, Soo-Hyun Choi, and Xia Hu. An information fusion approach to learning with instance-dependent label noise. In *International Conference on Learning Representations*, 2022.

[19] Nazmul Karim, Mamshad Nayeem Rizve, Nazanin Rahnavard, Ajmal Mian, and Mubarak Shah. Unicon: Combating label noise through uniform selection and contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9676–9686, 2022.

[20] Suyeon Kim, Dongha Lee, SeongKu Kang, Sukang Chae, Sanghwan Jang, and Hwanjo Yu. Learning discriminative dynamics with label corruption for noisy label detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22477–22487, 2024.

[21] Taehyeon Kim, Jongwoo Ko, Sangwook Cho, Jinhwan Choi, and Se-Young Yun. Fine samples for learning with noisy labels. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, pages 24137–24149, 2021.

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

[23] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2020.

[24] Shikun Li, Xiaobo Xia, Shiming Ge, and Tongliang Liu. Selective-supervised contrastive learning with noisy labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 316–325, 2022.

[25] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

[26] Yifan Li, Hu Han, Shiguang Shan, and Xilin Chen. Disc: Learning from noisy labels via dynamic instance-specific selection and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24070–24079, 2023.

[27] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342, 2020.

[28] Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, 2020.

[29] Yang Liu and Hongyi Guo. Peer loss functions: Learning from noisy labels without knowing noise rates. In *International conference on machine learning*, pages 6226–6236. PMLR, 2020.

[30] Leland McInnes, John Healy, and Steve Astels. hdbscan: Hierarchical density based clustering. *Journal of Open Source Software*, 2(11):205, 2017.

[31] Curtis Northcutt, Lu Jiang, and Isaac Chuang. Confident learning: Estimating uncertainty in dataset labels. *Journal of Artificial Intelligence Research*, 70:1373–1411, 2021.

[32] Weiran Pan, Wei Wei, Feida Zhu, and Yong Deng. Enhanced sample selection with confidence tracking: Identifying correctly labeled yet hard-to-learn samples in noisy data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 19795–19803, 2025.

[33] 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.

[34] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952, 2017.

[35] Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, 33:17044–17056, 2020.

[36] Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930, 2020.

[37] 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*, pages 8748–8763, 2021.

[38] Jialin Shi, Kailai Zhang, Chenyi Guo, Youquan Yang, Yali Xu, and Ji Wu. A survey of label-noise deep learning for medical image analysis. *Medical image analysis*, 95:103166, 2024.

[39] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. *Advances in neural information processing systems*, 32, 2019.

[40] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. Selfie: Refurbishing unclean samples for robust deep learning. In *International Conference on Machine Learning*, pages 5907–5915. PMLR, 2019.

[41] Daiki Tanaka, Daiki Ikami, Toshihiko Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5552–5560, 2018.

[42] Haobo Wang, Ruixuan Xiao, Yiwen Dong, Lei Feng, and Junbo Zhao. Promix: combating label noise via maximizing clean sample utility. *arXiv preprint arXiv:2207.10276*, 2022.

[43] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 322–330, 2019.

[44] Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 322–330, 2019.

[45] 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.

[46] Tong Wei, Hao-Tian Li, Chun-Shu Li, Jiang-Xin Shi, Yu-Feng Li, and Min-Ling Zhang. Vision-language models are strong noisy label detectors. *Advances in Neural Information Processing Systems*, 37:58154–58173, 2024.

[47] Jiayu Wu, Qixiang Zhang, and Guoxi Xu. Tiny imagenet challenge. *Technical report*, 2017.

[48] Pengxiang Wu, Songzhu Zheng, Mayank Goswami, Dimitris Metaxas, and Chao Chen. A topological filter for learning with label noise. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 21382–21393, 2020.

[49] Xiaobo Xia, Tongliang Liu, Bo Han, Mingming Gong, Jun Yu, Gang Niu, and Masashi Sugiyama. Sample selection with uncertainty of losses for learning with noisy labels. In *International Conference on Learning Representations*, 2022.

[50] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. Part-dependent label noise: Towards instance-dependent label noise. *Advances in neural information processing systems*, 33:7597–7610, 2020.

[51] Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. Part-dependent label noise: towards instance-dependent label noise. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, pages 7597–7610, 2020.

[52] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2691–2699, 2015.

[53] Yu Yao, Tongliang Liu, Mingming Gong, Bo Han, Gang Niu, and Kun Zhang. Instance-dependent label-noise learning under structural causal models. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, pages 4409–4420, 2021.

[54] Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7017–7025, 2019.

[55] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International conference on machine learning*, pages 7164–7173. PMLR, 2019.

[56] Shuo Zhang, Yuwen Li, Zhongyu Wang, Jianqing Li, and Chengyu Liu. Learning with noisy labels using hyperspherical margin weighting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 16848–16856, 2024.

[57] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[58] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in neural information processing systems*, 31, 2018.

[59] Zizhao Zhang, Han Zhang, Sercan O Arik, Honglak Lee, and Tomas Pfister. Distilling effective supervision from severe label noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9294–9303, 2020.

[60] Evgenii Zheltonozhskii, Chaim Baskin, Avi Mendelson, Alex M Bronstein, and Or Litany. Contrast to divide: Self-supervised pre-training for learning with noisy labels. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1657–1667, 2022.

[61] Zhaowei Zhu, Zihao Dong, and Yang Liu. Detecting corrupted labels without training a model to predict. In *International conference on machine learning*, pages 27412–27427. PMLR, 2022.

A Additional Ablations and Sensitivity Analyses

A.1 Sensitivity to the Cumulative Explained Variance Threshold λ in Class-wise Gradient Misalignment Denoising

We examine the sensitivity of GradTrack to the threshold λ introduced in Section 4.2, which controls the amount of cumulative explained variance retained by the denoised orthonormal basis. Figure 6 reports the average F1 score across all 11 benchmark settings, with error bars indicating standard deviations across the 11 settings, not uncertainty across repeated runs. Performance increases steadily as λ rises from 0.70 to 0.95, improving from 71.11% to 87.40%. However, at $\lambda = 1$, performance drops to 81.94%, suggesting that retaining the full spectrum introduces tail components dominated by noise, thereby reducing clean- and noisy-label separability.

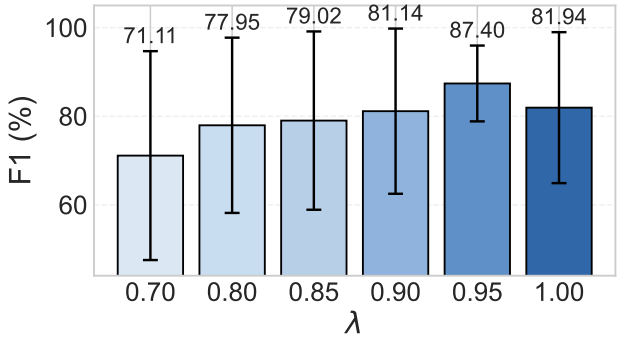


Figure 6: Average F1 across 11 benchmark settings for different cumulative explained variance thresholds λ. Error bars denote standard deviations across settings.

A.2 Over-cleansing Effect and Performance Sensitivity to the Cumulative Explained Variance Threshold ρ in Rank Trajectory Denoising

The CIFAR-10N Agg. setting illustrates a possible over-cleansing effect using the default 0.5 GMM posterior probability threshold. This setting differs from the other settings in two key aspects: 1) it has the lowest noise rate, and 2) it is one of the real-world noise settings. We believe that the observed over-cleansing effect is primarily associated with its low noise rate. At $\rho = 0.85$, while the AUC remains high (97.78%), the left-most confusion matrix in Table 3 translates to 92.81% recall but only 55.21% precision, resulting in an F1 of 69.23%. In addition, the detector flags 7,573 of 50,000 samples (15.15%) as noisy, whereas 4,505 labels (9.01%) are actually noisy. In comparison, across the other ten settings (including other real-world noise settings), recall is $96.59\% \pm 2.83\%$, and precision is $83.46\% \pm 10.46\%$ (mean $\pm$ standard deviation). Therefore, the low precision is likely the primary factor contributing to the low F1 score.

Table 3: Confusion matrix on CIFAR-10N Agg. with the default CLIP-based GradTrack configuration ($\rho \in \{0.85, 0.90, 0.95\}$, GMM posterior threshold 0.5). Entries are sample counts; $N = 50{,}000$.

ρ	0.85		0.90		0.95	
T (Retained components)	4		11		20	
	Predicted		Predicted		Predicted	
	Noisy label	Clean label	Noisy label	Clean label	Noisy label	Clean label
Actual noisy label	4,181	324	3,785	720	4,273	232
Actual clean label	3,392	42,103	1,080	44,415	13,230	32,265

Since GradTrack is designed to operate without prior knowledge of the noise rate, the purpose of this section is not to suggest tuning the GMM posterior threshold for low-noise settings merely. Instead, we investigate how other parameters in GradTrack may influence the extent of over-cleansing. The parameter ρ controls the amount of cumulative explained variance retained in the denoised rank trajectory before fitting the GMM (see Section 4.4). Figure 7 shows that the optimal value of ρ varies across settings. For the shown CIFAR-10 and CIFAR-100 settings, the default choice $\rho = 0.85$

remains optimal, while larger ρ values generally degrade performance. This is likely because retaining more components preserves additional rank inconsistencies, which weakens the separability between clean and noisy samples for the GMM. In contrast, CIFAR-10N Agg. exhibits a different trend, with a higher threshold of $\rho = 0.90$ achieving the best performance (80.79% F1). This suggests that the seven additional components retained when increasing ρ from 0.85 to 0.90 (Table 3) may capture additional signals that distinguish clean from noisy labels, before further components admit ranking inconsistencies that degrade the signal at $\rho = 0.95$, where a total of 20 components are retained.

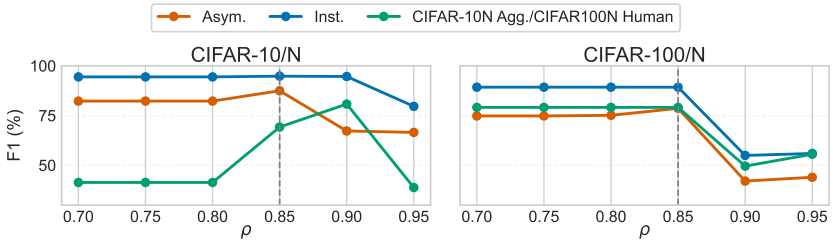


Figure 7: **Sensitivity to the explained-variance threshold ρ.** F1 as a function of ρ on representative CIFAR-10 and CIFAR-100 settings. The CIFAR-10 panel shows asymmetric, instance-dependent, and CIFAR-10N Agg. noise; the CIFAR-100 panel shows asymmetric, instance-dependent, and CIFAR-100N Human noise. The vertical dashed line marks the default choice $\rho = 0.85$.

We also note that the behavior observed on CIFAR-10N Agg. does not necessarily indicate a general failure pattern of the default configuration ($\rho = 0.85$) under real-world label noise, as GradTrack performs well on the other two real-world noise settings, CIFAR-10N Worst and CIFAR-100N Human, achieving 91.63% and 79.12% F1, respectively. To mitigate the over-cleansing effect, we suggest that when prior knowledge indicates a likely low noise rate and independent annotation is feasible, a small, randomly selected, manually verified subset could be used to find a suitable ρ value or GMM threshold. However, we have not empirically evaluated these extensions.

A.3 Sensitivity to Gradient Magnitude-based Sample Partitioning

We verify the gradient magnitude-based partition, which defines the low-, medium-, and high-magnitude groups. Figure 8-(a) reports the average F1 across all 11 benchmark settings for three partition ratios: 20–60–20, 25–50–25, and 33–33–33, effectively making the predominant clean group sample $\mathcal{S}^e_{l+m}$ with a decreasing 80%, 75%, and 66% of the entire training set. Performance is highly stable across these choices, with average F1 values of 87.30%, 87.38%, and 87.40%, respectively. This suggests that GradTrack does not depend strongly on a fine-grained partition ratio.

We further investigate the robustness of the partitioning strategies across different noise rates by considering instance-dependent noise with $\eta \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ on CIFAR-10, CIFAR-100, and Tiny-ImageNet. Table 4 reports the results for all 15 dataset–noise-rate combinations across the three partitioning strategies. The sample standard deviation in F1 across the three strategies is at most 1.05%, indicating that the performance is largely insensitive to the choice of partitioning strategy across different noise rates.

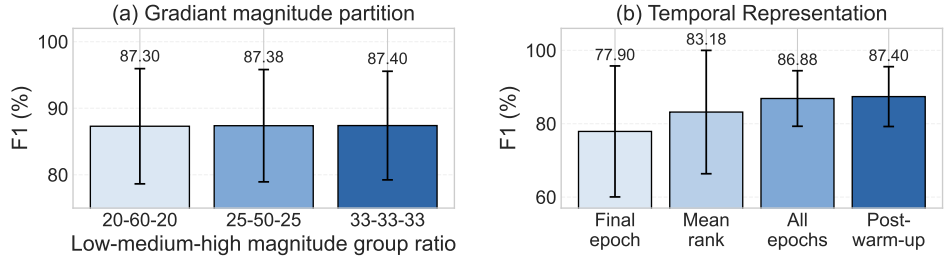


Figure 8: Average F1 across 11 benchmark settings (a) different low-, medium-, and high-magnitude partition ratios and (b) different temporal rank-trajectory aggregation strategies. Error bars denote the standard deviation across settings.

Table 4: F1 (%) for different low–medium–high magnitude partitions under instance-dependent noise. Mean and standard deviation (std) are computed across the three partitions in each row.

Dataset	η	20–60–20	25–50–25	33–33–33	mean $\pm$ std
CIFAR-10	0.1	87.62	88.09	88.20	87.97 ± 0.31
	0.2	91.36	91.24	91.30	91.30 ± 0.06
	0.3	94.23	94.23	94.22	94.23 ± 0.01
	0.4	94.78	94.79	94.80	94.79 ± 0.01
	0.5	85.58	85.56	85.56	85.57 ± 0.01
CIFAR-100	0.1	83.00	81.96	83.05	82.67 ± 0.62
	0.2	87.55	86.64	86.03	86.74 ± 0.76
	0.3	85.31	85.04	86.98	85.78 ± 1.05
	0.4	89.41	89.31	89.25	89.32 ± 0.08
	0.5	83.51	83.60	83.60	83.57 ± 0.05
Tiny-ImageNet	0.1	72.46	72.53	73.49	72.83 ± 0.58
	0.2	78.62	78.47	80.07	79.05 ± 0.88
	0.3	81.31	82.29	83.22	82.27 ± 0.96
	0.4	86.45	86.38	86.32	86.38 ± 0.07
	0.5	79.36	79.44	80.10	79.63 ± 0.41

A.4 Ablations on Temporal Rank Trajectory Aggregation

We examine the design choice described in Sec. 4.4: how temporal rank signals are selected and aggregated over the course of training. We consider three variants: (i) the last-epoch rank, (ii) the average rank across epochs, and (iii) the full rank trajectory over all epochs, and compare these variants to (iv) the proposed post-warmup rank trajectory. (iii) and (iv) are processed using SVD denoising followed by GMM to infer label noise, whereas (i) and (ii) are directly fed to GMM.

Figure 8-(b) shows a consistent improvement of (iii) and (iv) over (i) and (ii). This suggests that capturing the full extent of rank dynamics is more informative than relying on a single training snapshot, while signals from the warm-up phase are less informative.

A.5 Bottom-K for the Approximate Class-wise Clean-label Gradient

Table 5 reports the mean and standard deviation of F1 across the 11 benchmark settings for $K \in \{10, 25, 50, 100, 200, 500\}$. The best average performance is achieved at $K = 100$, followed closely by $K = 50$. Across the full tested range, mean F1 varies from 84.66% to 87.40%. The larger between-setting variability at $K = 200$ and $K = 500$ suggests that increasing K may introduce more hard clean and noisy samples into the bottom-K set, potentially degrading the estimation of the clean-label gradient direction.

Table 5: Sensitivity to K. Values are reported as mean F1 (%) $\pm$ standard deviation across the 11 benchmark settings, reflecting setting-wise variability rather than uncertainty across repeated runs.

	K	10	25	50	100	200	500
F1	Mean	85.73	85.23	87.15	**87.40**	85.28	84.66
	Standard deviation	9.53	10.40	8.60	8.16	13.85	12.78

A.6 GMM Alternatives

We compare the proposed two-component GMM with K-means and HDBSCAN [30] using the same denoised post-warmup rank-trajectory embeddings across all 11 settings. Table 6 shows that GMM achieves the highest mean F1 among the tested configurations. HDBSCAN automatically determines the number of clusters, with a minimum of two clusters specified in our experiment. It produces more than two clusters in 4 of the 11 settings and may leave some samples unclustered. In these cases, we identify the cluster with the highest mean as the predicted noisy group and assign unclustered samples to the predicted clean group. As a result, HDBSCAN's performance depends on the number of clusters formed and how unclustered samples are handled, which may have negatively affected

its performance in our experiments. Speedwise, using the CIFAR-10 Inst. 0.4 ($N = 50000$) setting as an example, GMM and K-means each require approximately 1.0–1.2 seconds, compared with approximately 23 seconds for HDBSCAN.

Table 6: Clustering method comparison. Values are reported as mean F1 (%) $\pm$ standard deviation across all 11 benchmark settings. All methods use the same trajectory embeddings.

Clustering methods		two-component GMM	K-means (K=2)	HDBSCAN
F1	Mean	**87.40**	81.26	59.48
	Standard deviation	8.16	18.45	18.01

B Empirical Validation of Observations on Gradient Dynamics

We extend the analysis in Sec. 3.2 from the illustrative CIFAR-100 Inst. 0.4 setting to all 11 noise detection settings in Table 1, using the frozen CLIP encoder. Oracle (true) class labels are used exclusively for this diagnostic evaluation. Table 7 summarizes the fractions of noisy labels, gradient alignment with the oracle clean-label gradient, and the signal correlation between gradient magnitude and misalignment signals. The reported standard deviations reflect variability across benchmark settings. We note that these empirical observations are specific to the tested settings and are not guaranteed to generalize to arbitrary datasets, encoders, or noise processes.

Table 7: Empirical validation across all 11 settings. All entries are the mean $\pm$ standard deviation across the settings. Spearman correlations are computed on the medium- and high-magnitude groups.

Diagnostic	Group or signal	All 11 settings	Exception subset
Noisy-label fraction (%)	Low	2.35 ± 2.76	—
	Medium	36.54 ± 27.58	—
	High	79.35 ± 22.90	—
Cosine with oracle gradient	Low	0.59 ± 0.23	0.87 ± 0.09
	Medium	0.60 ± 0.26	0.28 ± 0.32
	High	0.13 ± 0.32	0.01 ± 0.40
Spearman's rank correlation coefficient with magnitude (r_s)	Misalignment	0.53 ± 0.23	—
	Denoised misalignment	0.52 ± 0.22	—

Observation 1: Gradient magnitudes stratify clean and noisy label samples.

The low-magnitude group consistently contains fewer noisy-label samples than the medium- and high-magnitude groups across all 11 settings. This supports using gradient magnitude as a coarse indicator of clean- and noisy-label samples.

Observation 2: Low- and medium-magnitude groups provide useful clean-gradient approximations.

The medium-magnitude group exhibits the highest alignment with the oracle clean-label gradient in 8 of the 11 settings. The exceptions are CIFAR-10, CIFAR-100, and Tiny-ImageNet under symmetric noise at $\eta = 0.6$. In these three settings, noisy-label samples account for 76.6–79.9% of the medium-magnitude group, compared with 22.6–35.1% in the remaining settings. This high proportion of noisy samples may distort the average gradient direction of the medium-magnitude group, resulting in weaker alignment with the oracle clean-label gradient as compared to the low-magnitude group (a mean cosine similarity of 0.28 vs. 0.87; Table 7, exception subset column). Given the similar overall mean cosine similarities of 0.59 and 0.60 for the low- and medium-magnitude groups, respectively, including both groups is a reasonable choice for approximating the clean-label gradient direction.

Observation 3: Gradient misalignment provides complementary signals to gradient magnitude for detecting noisy labels.

We compute Spearman's rank correlation coefficient (r_s) to measure the correlation between gradient magnitude and the proposed misalignment signals over the post-warmup epochs from combined medium- and high-magnitude groups. We consider only the medium- and high-magnitude groups

because the low-magnitude group consists predominantly of clean-label samples, which are easily distinguished by both signals and may artificially inflate the r_s values. Across the 11 settings, $r_s = 0.53 \pm 0.23$ between gradient magnitude and class-wise misalignment signals, and 0.52 ± 0.22 between gradient magnitude and the denoised misalignment signals, suggesting the denoised signal is marginally less correlated with the gradient magnitude. However, a lower r_s indicates weaker rank correlation between the signals but does not necessarily imply better noisy-label detection performance. Table 8 shows that the gradient magnitude (**Magnitude-Only** column), class-wise gradient misalignment (**Baseline** column), and denoised misalignment (**+ $\mathcal{S}_{\text{l+m}}$** column) achieve average AUC scores of 85.22, 96.82, and 96.76, respectively, with the latter two exhibiting nearly identical empirical performance. The complementary roles of the gradient magnitude and gradient misalignment signals are supported by the stronger average F1 performance achieved by their joint trajectory (Table 8, **+Magnitude** column), 87.40% as compared to the individual 71.30% and 85.55% results.

Table 8: Ablation of GradTrack components across all 11 benchmark settings. Each cell reports F1 (%) with AUC in parentheses.

Setting	**Magnitude-Only**	**Baseline**	**+ All Gradients**	**+ $\mathcal{S}_{\text{l+m}}$**	**+ Magnitude**
CIFAR-10 Sym. 0.6	96.64 (99.06)	92.96 (99.59)	93.30 (99.64)	93.19 (99.65)	97.29 (99.32)
CIFAR-10 Asym. 0.3	39.80 (97.89)	66.01 (94.22)	76.48 (96.38)	80.44 (97.38)	87.48 (98.67)
CIFAR-10 Inst. 0.4	91.19 (97.88)	88.22 (98.87)	89.94 (98.93)	89.66 (98.93)	94.80 (99.13)
CIFAR-10N Worst 0.43	92.17 (98.42)	90.34 (97.52)	91.01 (98.00)	89.49 (97.68)	91.63 (98.80)
CIFAR-10N Agg. 0.093	70.26 (97.87)	54.02 (97.27)	65.64 (97.44)	69.65 (97.06)	69.23 (97.78)
CIFAR-100 Sym. 0.6	88.96 (92.31)	92.50 (98.72)	92.21 (98.26)	92.43 (98.19)	93.85 (98.05)
CIFAR-100 Asym. 0.3	38.72 (54.67)	88.86 (98.09)	89.29 (97.51)	88.36 (97.32)	78.65 (97.47)
CIFAR-100 Inst. 0.4	53.44 (61.46)	91.63 (98.63)	91.99 (98.25)	91.76 (98.36)	89.25 (98.75)
CIFAR-100N Human 0.43	77.24 (90.63)	60.28 (84.64)	63.32 (83.15)	61.52 (82.13)	79.12 (91.40)
Tiny-Imagenet Sym. 0.6	83.87 (86.27)	92.45 (98.53)	91.93 (98.47)	92.14 (98.43)	93.75 (97.82)
Tiny-Imagenet Inst. 0.4	52.05 (60.94)	91.98 (98.94)	92.32 (99.25)	92.39 (99.20)	86.32 (98.61)
Average	71.30 (85.22)	82.66 (96.82)	85.22 (96.84)	85.55 (96.76)	87.40 (97.80)

C Alternative Feature Encoder

Table 9: F1 (%) with a frozen ImageNet-pretrained ResNet-50 [13] encoder. The classification head is trained for 50 epochs in all settings.

Setting	SIMIFEAT-V [61]	GradTrack
CIFAR-10 Sym. 0.6	93.28	96.53
CIFAR-10 Asym. 0.3	65.94	63.99
CIFAR-10 Inst. 0.4	82.37	94.50
CIFAR-10N Worst 0.4	87.14	88.64
CIFAR-10N Agg. 0.09	66.94	43.05
CIFAR-100 Sym. 0.6	86.11	94.52
CIFAR-100 Asym. 0.3	68.19	78.36
CIFAR-100 Inst. 0.4	76.70	91.67
CIFAR-100N Human 0.4	75.25	73.60
Tiny-ImageNet Sym. 0.6	76.60	94.33
Tiny-ImageNet Inst. 0.4	76.72	86.96
Average	77.75	**82.38**

To investigate whether GradTrack's performance gains over the compared methods are specific to CLIP features, we evaluate GradTrack and SIMIFEAT-V using a frozen ImageNet-pretrained ResNet-50 [13] across the same 11 settings, with the classification head trained for 50 epochs in each setting. As shown in Table 9, GradTrack achieves an average F1 of 82.38%, outperforming SIMIFEAT-V at 77.75%.

Training setting variations. We note that the ResNet-50 experiments use different configurations from those with CLIP in two aspects. 1) The cumulative explained variance threshold is set to $\rho = 0.7$, as higher values of ρ appear to introduce more harmful ranking inconsistencies in ResNet-50

features than in CLIP features. 2) The number of training epochs is increased from 30 to 50 for the CIFAR-10 experiments, as the classification head does not appear to converge within the 30 epochs used for the corresponding CLIP experiments.

Failure case analysis and limitations. GradTrack underperforms SIMIFEAT-V on the low-noise CIFAR-10N Agg. setting (43.05% vs. 66.94% F1). Despite achieving a high AUC of 96.38% and recall of 96.98%, its precision is only 27.66%, with 31.59% of samples predicted as noisy against a true noise rate of 9.01%. This indicates that strong ranking performance can still coexist with substantial over-cleansing, which is more severe than that observed with CLIP. GradTrack also slightly underperforms SIMIFEAT-V on CIFAR-10 Asym. 0.3 (63.99% vs. 65.94%) and CIFAR-100N Human (73.60% vs. 75.25%). Overall, these results demonstrate that GradTrack's average performance gains extend beyond CLIP features, while highlighting that the over-cleansing issue persists across different encoders in low-noise settings.

D Computational Cost and Practical Scope

Table 10 profiles the runtime of GradTrack's processing stages on Tiny-ImageNet with instance-dependent noise at $\eta = 0.4$. The dataset contains 100,000 training images across 200 classes, and the model is trained for 50 epochs using the implementation described in Sec. 5.1. The total runtime is 10,877.5 seconds (approximately 3 hours), with per-sample gradient extraction accounting for 98.97% of the total runtime. All subsequent GradTrack processing stages collectively require only 112.5 seconds (1.03%). Thus, the primary computational bottleneck of GradTrack is per-sample gradient extraction, a requirement shared with other gradient-based detection methods such as TracIn [36].

The relatively small overhead of the projection, subspace, and clustering stages does not eliminate the computational cost of per-sample gradient extraction. Nevertheless, the total runtime of approximately 3 hours for 100,000 training samples demonstrates the feasibility of GradTrack for offline noise detection or periodic dataset filtering at this scale. We note that these measurements are based on a single dataset and implementation, and do not establish scalability to substantially larger datasets or models. Peak memory usage and computational scaling with larger class counts or full-model gradients remain to be evaluated.

Table 10: Runtime breakdown on Tiny-ImageNet Inst. 0.4 over 50 epochs. Percentages use the sum of the listed stages as the denominator. One-off stages are applied after collecting the trajectories.

Stage	Per epoch (s)	Total (s)	Share (%)
Gradient extraction	215.3	10,765.0	98.97
Processes after gradient extraction	2.19	112.5	1.03
-Gradient partitioning	0.066	3.3	0.030
-Class-wise clean-label direction estimation	0.15	7.5	0.069
-SVD subspace construction	0.77	38.5	0.35
-Denoised misalignment computation	1.20	60.0	0.55
-Temporal rank modeling (One-off)	—	1.3	0.012
-GMM isolation (One-off)	—	1.9	0.017
All listed stages	—	10,877.5	100.00

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses the work's limitations.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The paper does not present formal theorems, lemmas, or proofs.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper includes detailed descriptions of the experimental setup and parameters, with additional ablations.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The paper uses public datasets. The paper will provide code upon acceptance.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

Answer: [Yes]

Justification: See Section 5.1 for details about training and test details.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: The main benchmark tables report single numbers without repeated-run uncertainty or statistical significance tests. The appendix error bars reflect variation across settings.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The computational resources utilized are described in Section 5.1.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper includes a broader impact discussion in Section 6.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The licenses for existing assets are properly cited and respected.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not introduce a new dataset, benchmark, or released model asset.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research?

Answer: [N/A]

Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components. LLMs were used only for grammar correction and refinement.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.