FROM SELECTION TO EXPOSURE: ONLINE DATA SELECTION VIA LEARNING-SIGNAL PROPAGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Data selection aims to reduce the cost of training by allocating computation toward informative examples, while maintaining comparable performance as example utility changes throughout learning. Existing methods estimate sample importance using early-training statistics, losses, or learned selection policies and then prune or prioritize a subset of the training data. While effective, these approaches can over-concentrate training on currently prioritized samples, reducing coverage of other regions of the training distribution. We propose an online data-selection framework that turns each mini-batch into feedback for selecting subsequent training examples. At each optimization step, feedback from the current batch continuously reshapes the sampling distribution over nearby examples in the evolving representation space, increasing exposure to currently informative regions without deterministically excluding lower-priority samples. This allows the current batch to directly influence future training exposure while preserving broader data coverage and reusing representations naturally produced during training. Across image classification, semantic segmentation and instruction tuning, SPADE achieves strong predictive performance with high training efficiency. On Tiny-ImageNet, SPADE reaches comparable accuracy with up to 67.2% fewer training sample presentations and 64.1% less wall-clock time. Across the three image-classification benchmarks, it achieves the best Top-1 accuracy in 12 of the 15 settings evaluated. On ADE20K, SPADE reaches 42.0% mIoU using 70% of the training data, improving over the compared vanilla baseline. For LLaMA-7B instruction tuning using only a small fraction of Alpaca, SPADE also improves over the compared baselines, achieving the highest average score of 26.5 across BBH, DROP, MMLU and HumanEval.

1 INTRODUCTION

Modern deep learning increasingly relies on large-scale datasets, but processing every available example throughout training is computationally expensive and often inefficient. Large datasets also contain substantial redundancy, noisy or atypical samples and examples whose usefulness changes as the model learns. These issues have motivated a broad class of *data-selection* methods that aim to allocate training computation toward examples that provide greater learning value.

Static approaches estimate sample utility using forgetting statistics, gradient- or prediction-error-based scores, geometric coverage and sample difficulty (Toneva et al., 2018; Paul et al., 2021; Sener & Savarese, 2017; Xia et al., 2023). Methods such as CCS, D^2-Pruning and InfoMax additionally account for coverage, neighborhood structure, or inter-sample redundancy (Zheng et al., 2023; Maharana et al., 2023; Tan et al., 2025). Dynamic methods instead adapt sample priorities throughout training. Online Batch Selection uses recent losses (Loshchilov & Hutter, 2015), InfoBatch dynamically reduces the exposure of low-loss samples (Qin et al., 2024) and Data Agent learns a PPO-based selection policy from difficulty- and uncertainty-related signals (Yang et al., 2026). A broader discussion of related data-selection methods is provided in Appendix D.

Despite this progress, existing methods still face two limitations. First, many approaches ultimately make relatively hard keep-or-discard decisions. Samples judged useful are retained or repeatedly prioritized, while easier or lower-scoring samples may be excluded. However, easy samples are not necessarily useless: they can provide stable learning signals and preserve already learned structure.

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

Over-concentrating the training budget on difficult examples may therefore disturb the balance between consolidating existing knowledge and acquiring new information.

Second, selection is often performed before training or refreshed only at epoch boundaries. This creates a temporal mismatch between the model state used to estimate sample utility and the state at which the selected data are consumed. Consequently, sample relevance can change within the same epoch: examples that are informative at early optimization steps may become redundant later, while previously easy or low-priority examples may become useful again as the decision boundary evolves.

Motivated by these limitations, we introduce **SPADE: Semantic Propagation for Adaptive Data Exposure**. Rather than constructing a hard subset, our method maintains a time-varying sampling distribution over the training set. At each optimization step, the current mini-batch provides a lightweight per-example learning signal, which is propagated through representation-space neighborhoods to adjust the sampling probabilities of related samples. Higher-signal regions receive more training budget, while lower-priority samples are down-weighted rather than removed.

This design has two key properties: it is *responsive*, because sampling priorities are updated at mini-batch frequency and it is *soft*, because all samples remain reachable. In this way, the method aims to prioritize informative examples without permanently excluding easier ones.

Our contributions are:

- We propose **SPADE**, a step-level data-selection framework that continuously adapts the sampling distribution as the model evolves.
- We introduce a semantic learning-signal propagation mechanism that transfers optimization feedback from currently observed examples to related unobserved candidates.
- Extensive experiments across multiple datasets and model architectures show that SPADE consistently outperforms competitive data-selection approaches under matched training budgets, while reducing training sample usage by over 60% without sacrificing predictive performance.

2 METHOD

Problem formulation. Given a training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i, y_i are the input and label of sample i, we formulate data selection as a time-varying sampling policy $\pi = \{P_t\}_{t=1}^{T}$. At step t, P_t defines a probability distribution over the currently available samples and is used to draw the mini-batch B_t.

We seek a policy that minimizes the final-model risk under a fixed training budget K:

$$\begin{aligned} \min_{\pi} \quad & \mathbb{E}_{B_{1:T}\sim\pi}\left[\mathcal{R}(\theta_T)\right] \\ \text{s.t.} \quad & \theta_{t+1} = \theta_t - \eta_t \frac{1}{|B_t|} \sum_{i\in B_t} \nabla_\theta L_i(\theta_t), \\ & B_t \sim P_t, \qquad \sum_i P_t(i) = 1, \\ & \sum_{t=1}^{T} |B_t| \le K. \end{aligned} \tag{1}$$

Here, θ_t is the model state at step t, η_t is the learning rate, L_i is the loss of sample i, $|B_t|$ is the batch size, $\mathcal{R}(\theta_T)$ is the final-model risk and K is the maximum number of sample uses during training.

Figure 1 summarizes the overall training loop, from per-sample learning-signal estimation to neighborhood propagation and adaptive reweighting of future training exposure.

2.1 WHAT MAKES AN EXAMPLE WORTH TRAINING ON?

Previous work has shown that various signals can be used to identify which samples are worth prioritizing during training. Data Agent (Yang et al., 2026) uses loss-based difficulty and model

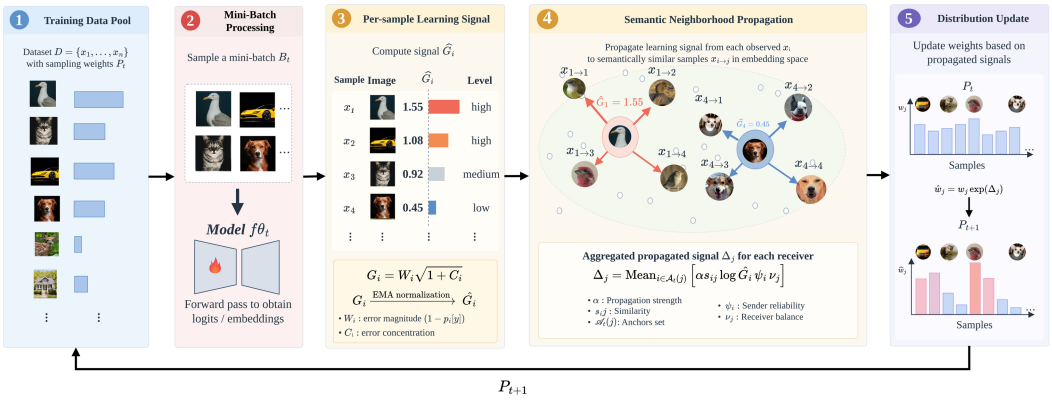

Figure 1: Overview of SPADE. At each training step, a mini-batch is sampled from the current distribution and processed by the task model to obtain logits and embeddings. Per-sample learning signals are computed and normalized, then propagated to semantically related candidates through representation-space neighborhoods. The aggregated propagated signal updates receiver priorities, which are converted into the next sampling distribution P_{t+1}, forming a closed-loop adaptive data exposure process. Sample images used for illustration are retrieved from publicly available Kaggle datasets (markovka; Ayman; Shanbalico; klu2000030172).

confidence, InfoBatch (Qin et al., 2024) relies on per-example loss and InfoMax (Tan et al., 2025) combines sample-wise informativeness, such as loss, with inter-sample similarity for diverse coreset selection. Building on these methods, we capture both example difficulty and the structure of competing predictions.

Difficulty signal. We leverage the model's probability assigned to the target output as a target-confidence-based proxy for the current difficulty of each example. For a classification example, let $p_i[y_i]$ denote the probability assigned to the target class. We define

$$W_i = 1 - p_i[y_i]. \tag{2}$$

The signal is obtained directly from the standard forward pass and evolves naturally throughout training and is architecture-agnostic because it depends only on the model output and target rather than on a specific network design.

Concentration signal. The difficulty signal reflects how well the current model fits an example, but it does not capture how strongly the remaining probability mass is concentrated among alternative outputs. We therefore introduce a concentration signal that characterizes the distribution of non-target probability mass across competing classes.

$$C_i = \frac{\sum_{k \neq y_i} p_i[k]^2}{W_i^2}, \tag{3}$$

where k runs over all non-target classes. Following the form of the Herfindahl-Hirschman concentration index (Herfindahl, 1950; Hirschman, 1964), C_i quantifies how concentrated the total incorrect probability mass W_i is across competing classes. A larger C_i indicates that the non-target mass is concentrated on a small number of strong competitors, whereas smaller values correspond to more diffuse confusion.

The two signals capture complementary aspects of the current prediction: W_i measures how poorly the example is fitted, while C_i characterizes the structure of competing predictions. We combine them into the learning signal

$$G_i = W_i\sqrt{1 + C_i}. \tag{4}$$

This form also admits a direct optimization interpretation. For softmax cross-entropy, the derivative of the per-example loss with respect to the logits is simply the prediction error,

$$\frac{\partial L_i}{\partial o_i} = p_i - q_i. \tag{5}$$

Consequently,

$$G_i = \left\| \frac{\partial L_i}{\partial o_i} \right\|_2, \tag{6}$$

so G_i measures the magnitude of the optimization signal induced by an example at the model output. This interpretation is closely connected to Error L2-Norm (Paul et al., 2021).

Relative learning signal. The scale of G_i changes throughout training, typically decreasing as the model becomes better fitted. Consequently, the same absolute value can have different meanings at different stages of optimization. We therefore normalize G_i by an exponential moving average of the recent batch mean,

$$\bar{G}_t = \beta \bar{G}_{t-1} + (1-\beta) \frac{1}{|B_t|} \sum_{i \in B_t} G_i, \tag{7}$$

and define

$$\hat{G}_i = \frac{G_i}{\bar{G}_t}. \tag{8}$$

2.2 From Learning Signals to Training Exposure

Because G_i is available only for samples observed in the current mini-batch, we propagate it to nearby examples in the model representation space Yang et al. (2026). We maintain a persistent cache of ℓ_2-normalized features $\hat{z}_i$ and for each anchor $i \in B_t$, we retrieve up to k nearest neighbors from the remaining candidates in the current epoch,

$$\mathcal{N}_t(i) = \underset{j \in \mathcal{C}_t}{\text{top-k}}\, \hat{z}_i^\top \hat{z}_j, \tag{9}$$

The cache is refreshed whenever a sample is revisited, allowing neighborhoods to evolve with the model without additional encoder evaluations.

The non-negative similarity s_{ij} determines how strongly the learning signal of anchor i is transferred to a neighboring candidate j. We define the corresponding propagation signal as

$$\ell_{i \to j} = \alpha\, s_{ij}\, \log \hat{G}_i\, \psi_i\, \nu_j, \tag{10}$$

where α controls the overall propagation strength, ψ_i and ν_j calibrate the contribution of the sender and receiver, respectively.

When a candidate j is reached by multiple anchors in the current batch, their signals are averaged. This prevents candidates that appear in many anchor neighborhoods from receiving an excessively large update simply because they accumulate numerous weak propagation signals. Let $\mathcal{A}_t(j)$ denote the anchors whose retrieved neighborhoods contain candidate j.

$$\Delta_j = \frac{1}{|\mathcal{A}_t(j)|} \sum_{i \in \mathcal{A}_t(j)} \ell_{i \to j}, \tag{11}$$

and its persistent sampling priority is updated multiplicatively,

$$w_j \leftarrow w_j \exp(\Delta_j). \tag{12}$$

This exponentiated update preserves positive priorities while translating additive evidence into proportional changes in future exposure (Kivinen & Warmuth, 1997). Importantly, the update does not force a sample to be selected or discarded; it only changes its probability of appearing in future mini-batches.

Sender reliability. Not every retrieved neighborhood provides equally strong evidence that an anchor's signal should generalize to nearby candidates. We quantify the effective support of the neighborhood using the weighted effective sample size of Kish (1965),

$$n_i^{\text{eff}} = \frac{\left(\sum_j s_{ij}\right)^2}{\sum_j s_{ij}^2}. \tag{13}$$

A neighborhood supported by fewer candidates has a smaller effective size, whereas broader support produces a larger value. We map this statistic to a smooth reliability coefficient,

$$\psi_i = \exp\left[-\frac{1}{2}\left(\frac{\log n_i^{\text{eff}} - \mu}{\varsigma}\right)^2\right], \tag{14}$$

where μ and ς characterize the empirical distribution of $\log n^{\text{eff}}$. This term moderates propagation from anchors whose local neighbourhood provides atypical support.

Receiver balance. Candidates also differ in how frequently they are retrieved. Without correction, highly connected examples may accumulate larger updates simply because they appear in more neighbourhoods. Inspired by degree normalization in graph propagation (Kipf & Welling, 2017), we define

$$\nu_j = \frac{\bar{d}_t}{\max(\deg_t^-(j), 1)}, \qquad \bar{d}_t = \frac{1}{N}\sum_{j=1}^{N} \deg_t^-(j), \tag{15}$$

where $\deg_t^-(j)$ is the current number of incoming neighbourhood connections. This reduces connectivity-driven amplification so that priority changes depend more strongly on propagated learning evidence than on retrieval frequency alone.

2.3 Adaptive Training Exposure

The propagated updates are accumulated in a positive priority weight w_j, which parameterizes the probability that example j will receive future training exposure. A higher weight increases expected exposure, while a lower weight reduces it without removing the example from the sampling support.

Because the relevance of propagated evidence changes as the model evolves, we gradually return priorities toward the uniform reference,

$$\tilde{w}_j \leftarrow (1-\rho)w_j + \rho, \tag{16}$$

where ρ controls the recovery rate. This prevents transient learning signals from permanently determining future exposure.

The resulting sampling distribution is

$$P_t(j) = \frac{\tilde{w}_j \, \mathbf{1}[j \in \mathcal{C}_t]}{\sum_{k \in \mathcal{C}_t} \tilde{w}_k}. \tag{17}$$

Batches are drawn without replacement from P_t. Hence, propagated learning signals continuously redistribute training exposure: currently informative regions receive more computation, while lower-priority examples remain reachable and can regain exposure as the model state changes.

2.4 TRAINING OBJECTIVE

Adaptive exposure changes how frequently individual examples contribute to optimization. To compensate for unequal sampling probabilities, we apply an inverse-probability correction motivated by unequal-probability sampling (Horvitz & Thompson, 1952),

$$\mathcal{L}_t = \frac{1}{|B_t|} \sum_{i \in B_t} a_i L_i, \qquad a_i = \min\left(\frac{1}{|\mathcal{C}_t| P_t(i)}, c \right), \tag{18}$$

where c clips excessively large corrections caused by very small sampling probabilities. Under uniform sampling over $\mathcal{C}_t$, $a_i = 1$ and the objective reduces to the ordinary batch mean.

Because sampling is performed without replacement, $P_t(i)$ is a pre-draw probability rather than the exact inclusion probability required by an unbiased Horvitz-Thompson (Horvitz & Thompson, 1952) estimator. Clipping further introduces deliberate bias in exchange for reduced variance. We therefore use Equation 18 as an importance-weighted scale correction rather than claiming unbiased estimation of the full-data objective.

3 EXPERIMENTS

We evaluate data selection under fixed training budgets across several settings: image classification, semantic segmentation and instruction tuning. For each task, we vary the fraction of training examples presented to the model and report predictive performance. We first conduct a broad comparison on image classification to identify the most competitive methods in this setting in terms of predictive performance. Based on these results, we carry forward InfoBatch (Qin et al., 2024) and Data Agent (Yang et al., 2026) for semantic segmentation and InfoBatch (Qin et al., 2024) for instruction tuning. Due to limited computational resources, we restrict these additional cross-task comparisons to these selected baselines. All the implementation details and hyperparameters setup are listed in Appendix A. This evaluation measures both task performance and data efficiency while accounting for differences in selection schedules and effective training budgets across methods.

3.1 IMAGE CLASSIFICATION

We train ResNet-18 (He et al., 2016) from scratch on CIFAR-10 and CIFAR-100 (Krizhevsky & Hinton, 2009), using the same setup with InfoMax (Tan et al., 2025). For Tiny-ImageNet (Wu, 2017), we use a torchvision ResNet-50 (He et al., 2016) with the detail setup listed in A. Top-1 accuracy is evaluated on the complete held-out split.

We compare SPADE against a broad set of static and dynamic data-selection baselines. Among static methods, Random provides a uniform-sampling reference, while Herding (Welling, 2009) greedily selects examples to approximate the feature mean of the full dataset. Forgetting (Toneva et al., 2018) ranks examples by the number of correct-to-incorrect prediction transitions during training, whereas EL2N (Paul et al., 2021) measures the ℓ_2 distance between the predicted probability vector and the target label. K-Center (Sener & Savarese, 2017) selects examples to improve geometric coverage of the representation space, while Moderate-DS (Xia et al., 2023) favors examples whose distance-to-class-center scores lie near the median. CCS (Zheng et al., 2023) stratifies samples by importance score and samples across score ranges to improve coverage. D^2-Pruning (Maharana et al., 2023) propagates difficulty information through a data graph to account for both difficulty and diversity, while InfoMax (Tan et al., 2025) combines sample-wise informativeness with pairwise similarity to favor informative yet non-redundant coresets.

For dynamic selection, Random* periodically resamples training subsets, we use this as a baseline for dynamic pruning methods, ϵ-Greedy and UCB (Raju et al., 2021) use dynamically updated sample-loss scores with different exploration strategies. InfoBatch (Qin et al., 2024) dynamically prunes lower-loss examples and reweights the retained samples, while Data Agent (Yang et al., 2026) learns a PPO-based selection policy from difficulty and uncertainty signals.

Table 1 shows that our method is consistently competitive with or better than prior data-selection approaches across the three image-classification benchmarks. The gains are most consistent on

Table 1: A comparative analysis of Top-1 accuracy (%) under different data-selection ratios on CIFAR-10, CIFAR-100 and Tiny-ImageNet. The best results are bolded. Detailed result sources and reproduction settings are provided in the Appendix A

Type	Method	CIFAR-10					CIFAR-100					Tiny-ImageNet				
	Whole Dataset	95.6					78.7					47.9				
	Selection Rate	**10%**	**20%**	**30%**	**50%**	**70%**	**10%**	**20%**	**30%**	**50%**	**70%**	**10%**	**20%**	**30%**	**50%**	**70%**
Static	Random	79.0	88.0	90.9	93.4	94.3	44.8	57.4	65.3	71.1	74.6	15.9	26.0	29.8	37.0	41.9
	Herding (Welling, 2009)	82.6	88.5	80.1	88.0	94.1	46.6	48.4	50.9	73.5	73.4	19.7	27.5	31.2	38.4	43.6
	Forgetting (Toneva et al., 2018)	46.3	73.8	88.1	94.9	**95.7**	20.6	35.6	49.8	68.1	76.0	19.8	28.1	33.3	41.4	45.1
	EL2N (Paul et al., 2021)	30.3	78.6	89.2	94.8	95.4	11.8	27.3	42.0	68.1	75.6	3.8	19.8	26.6	36.9	44.1
	K-Center (Sener & Savarese, 2017)	68.4	82.5	91.0	92.5	94.1	40.5	47.2	61.4	69.5	73.4	5.0	12.4	20.5	33.2	41.1
	Moderate-DS Xia et al. (2023)	81.0	87.3	90.6	92.6	93.9	45.5	51.8	57.8	71.1	74.6	14.6	25.1	30.6	35.4	44.4
	CCS (Zheng et al., 2023)	86.9	91.0	93.0	95.0	95.4	57.3	64.0	68.9	74.4	77.1	19.1	25.1	30.2	35.4	40.9
	D^2-Pruning Maharana et al. (2023)	87.1	91.4	93.3	94.9	**95.7**	56.9	65.2	70.5	75.9	78.2	19.2	28.6	32.9	41.3	45.0
	InfoMax Tan et al. (2025)	89.1	92.7	94.1	94.7	95.5	58.7	67.9	71.5	76.7	79.0	5.3	10.9	18.2	28.0	36.3
Dynamic	Random* (Raju et al., 2021)	87.2	92.1	93.0	94.5	94.8	65.8	73.0	74.9	75.3	77.3	23.9	35.6	41.8	45.6	46.5
	ϵ-Greedy (Raju et al., 2021)	85.5	93.9	94.1	94.9	95.2	65.1	73.5	74.1	74.8	76.4	8.6	27.9	33.8	43.5	45.6
	UCB (Raju et al., 2021)	86.5	93.7	93.9	94.7	95.3	59.7	73.1	74.4	75.3	77.3	14.7	28.5	30.3	40.9	44.4
	InfoBatch Qin et al. (2024)	89.4	94.1	**94.9**	95.2	95.3	62.7	75.7	76.5	78.1	79.0	16.9	37.6	43.0	46.7	46.8
	Data Agent (Yang et al., 2026)	91.2	93.4	94.3	94.7	95.3	67.3	74.2	75.6	77.4	78.0	**31.1**	38.5	41.1	42.9	47.4
	SPADE	**91.4**	**94.3**	94.8	**95.3**	95.4	**71.9**	**76.4**	**77.5**	**78.2**	**79.2**	25.9	**42.9**	**44.9**	**47.1**	**48.2**

Table 2: Segmentation mIoU (%) on ADE20K using UPerNet Xiao et al. (2018).

Selection Rate	**70%**	**100%**
Baseline (full data)	–	40.8
Data Agent	41.5	–
InfoBatch	41.1	–
SPADE	**42.0**	–

CIFAR-100, where our method performs best across all reported selection ratios and on Tiny-ImageNet, where it performs best from 20% selection onward. On CIFAR-10, our method remains competitive across all budgets and achieves the strongest results at several selection ratios. Overall, the results indicate that the proposed online exposure mechanism is particularly effective under reduced training budgets, especially on the more challenging CIFAR-100 and Tiny-ImageNet benchmarks.

3.2 Semantic Segmentation

We examine transfer to dense prediction on ADE20K (Zhou et al., 2016). At the 70% selection ratio, our method achieves the highest mIoU of 42.0%, outperforming Data Agent (Yang et al., 2026) by 0.5 percentage points and InfoBatch (Qin et al., 2024) by 0.9 percentage points. It also exceeds the full-data random baseline of 40.8%, showing that the proposed selection strategy can retain and in this setting improve, segmentation performance while using only 70% of the training data.

3.3 Instruction Tuning for Large Language Model

We evaluate instruction tuning of LLaMA-7B (Touvron et al., 2023) on the Alpaca dataset (Taori et al., 2023) and report performance on four downstream benchmarks: BBH (Suzgun et al., 2023), DROP (Dua et al., 2019), MMLU (Hendrycks et al., 2020) and HumanEval (Chen et al., 2021). Following the 2% data-selection setting used by Data Quantization (Zhou et al., 2023) and the InfoBatch evaluation (Qin et al., 2024), we compare methods under the same selection ratio. As shown in Table 3, SPADE achieves the highest scores on DROP, MMLU and HumanEval, as well as the best average score of 26.5, demonstrating that the proposed exposure mechanism also transfers effectively to instruction tuning.

3.4 Experiments on noisy, corruption dataset

We additionally evaluate robustness to training-set corruption using the same corruption protocol as Clip-Sel Yang et al. (2025). On Tiny-ImageNet Wu (2017), we consider 20% label noise and

Table 3: Instruction tuning of LLaMA-7B (Touvron et al., 2023) on Alpaca (Taori et al., 2023), evaluated on downstream tasks.

Method	BBH	DROP	MMLU	Human-Eval	Avg.
DQ+InfoBatch	31.0	27.5	35.3	**11.6**	26.3
DQ (2% data)	**32.9**	27.6	36.6	8.5	26.3
SPADE (r = 2%)	28.3	**29.0**	**36.9**	**11.6**	**26.5**

Table 4: Best validation accuracy (%) on Tiny-ImageNet under 20% label noise and 20% visual corruption.

Condition	Method	30%	50%	70%	100%
Label noise	Baseline (full data)	–	–	–	36.0
	Data Agent	32.7	34.9	36.7	–
	InfoBatch	38.8	**38.7**	37.2	–
	SPADE	**39.2**	38.4	**38.0**	–
Visual corruption	Baseline (full data)	–	–	–	45.8
	Data Agent	37.9	40.7	44.3	–
	InfoBatch	**46.1**	**46.9**	40.4	–
	SPADE	43.6	46.2	**46.9**	–

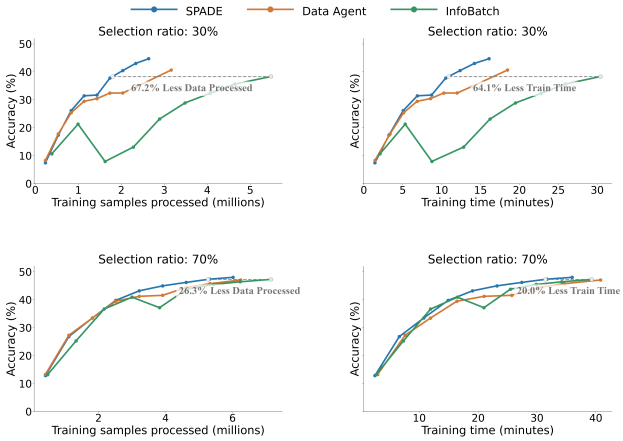

Figure 2: Training efficiency comparison on Tiny-ImageNet at 30% and 70% selection ratios, measured by accuracy versus cumulative training samples and wall-clock training time.

20% visual corruption, with the corrupted images distributed equally across fog, Gaussian noise, motion blur, occlusion and resolution degradation. Under 20% label noise, our method achieves the best accuracy at 30% and 70% selection, reaching 39.16% and 37.99%, respectively, while remaining competitive with InfoBatch Qin et al. (2024) at 50%. Under 20% visual corruption, our method performs best at the 70% selection ratio with 46.92% accuracy and remains competitive at 50%, while InfoBatch Qin et al. (2024) performs better at 30%. Overall, the results indicate that the proposed selection strategy remains robust under both noisy labels and visual corruption, with stronger performance particularly at moderate-to-high selection ratios.

3.5 Training Efficiency

Figure 2 compares accuracy as a function of cumulative training samples and wall-clock training time on Tiny-ImageNet. Since Data Agent (Yang et al., 2026) and InfoBatch (Qin et al., 2024) are designed to reduce training cost relative to full-data training, we focus on their relative efficiency under comparable selection budgets rather than repeating comparisons against the vanilla full-data baseline. At the 30% selection ratio, SPADE reaches comparable accuracy while processing up

to 67.2% fewer training examples and requiring 64.1% less wall-clock time than the competing methods. At the 70% ratio, the reductions remain 26.3% in processed data and 20.0% in training time. These results show that SPADE improves not only predictive performance but also the rate at which useful accuracy is obtained from a limited training budget, with larger gains under more constrained selection ratios.

4 CONCLUSION

We introduced SPADE, an online data-selection framework that adapts training exposure at mini-batch granularity by propagating fresh learning signals through the model's evolving representation space. Instead of constructing a fixed subset, SPADE maintains a dynamic sampling distribution, allowing informative regions to receive more computation while lower-priority examples remain reachable throughout training. The proposed learning signal is derived directly from the model outputs and the propagation mechanism reuses representations already produced during training, avoiding auxiliary selection models or separate scoring networks. Across image classification, semantic segmentation and instruction tuning, the method consistently achieves competitive or improved performance under reduced training budgets and remains robust under label noise and visual corruption. These results support adaptive exposure as a promising alternative to hard data pruning, while future work will focus on scaling neighborhood retrieval to substantially larger datasets and further improving the efficiency of the online controller.

Limitations and future work. Our method currently lacks a principled rule for selecting the neighborhood size k. Empirically, we observe that an appropriate value of k is related to the dataset size N, batch size b and selection ratio r. However, we have not yet established a general formulation that precisely characterizes this relationship across different datasets and training configurations, see Appendix B for further details. Developing a more principled and broadly applicable rule for choosing k is an important direction for future work.

Our method also relies on the quality of the representations used for neighborhood retrieval. When training is restricted to a limited number of epochs, the target model may not develop sufficiently informative embeddings, making the retrieved neighborhoods less reliable. In this setting, a pretrained embedding model can provide an alternative representation space, as discussed in Appendix B. Adapting the method to such short training schedules, particularly in large-scale pretraining, remains an important direction for future work.

AI USE STATEMENT

Generative AI (OpenAI Codex) was used to convert the supplied draft to the ICLR 2027 LaTeX format, assist with editing, reference formatting and identify inconsistencies in the draft.

REFERENCES

Samuel Ayman. Cat dataset. https://www.kaggle.com/datasets/samuelayman/cat-dataset. Kaggle dataset, accessed September 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2368–2378, 2019.

Ruiqi Guo, Philip Sun, Erik Lindgren, Quan Geng, David Simcha, Felix Chern, and Sanjiv Kumar. Accelerating large-scale inference with anisotropic vector quantization. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pp. 3887–3896. PMLR, 2020. URL https://proceedings.mlr.press/v119/guo20h.html.

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

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Orris C. Herfindahl. *Concentration in the Steel Industry*. PhD thesis, Columbia University, 1950.

Albert O. Hirschman. The paternity of an index. *American Economic Review*, 54(5):761–762, 1964.

D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *Journal of the American Statistical Association*, 47(260):663–685, 1952.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Leslie Kish. *Survey Sampling*. John Wiley & Sons, 1965.

Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

klu2000030172. Birds image dataset. https://www.kaggle.com/datasets/klu2000030172/birds-image-dataset. Kaggle dataset, accessed September 2026.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *Workshop track - ICLR 2016*, 2015. URL https://arxiv.org/abs/1511.06343.

Adyasha Maharana, Prateek Yadav, and Mohit Bansal. D2 pruning: Message passing for balancing diversity and difficulty in data pruning. *arXiv preprint arXiv:2310.07931*, 2023.

Yu. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(4):824–836, 2020. doi: 10.1109/TPAMI.2018.2889473.

markovka. Dataset of color cars. https://www.kaggle.com/datasets/markovka/car-dataset. Kaggle dataset, accessed September 2026.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 20596–20607. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/ac56f8fe9eea3e4a365f29f0f1957c55-Paper.pdf.

Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Zhaopan Xu, Daquan Zhou, Lei Shang, Baigui Sun, Xuansong Xie, and Yang You. InfoBatch: Lossless training speed up by unbiased dynamic data pruning. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2303.04947.

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

Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *International Conference on Learning Representations, arXiv preprint arXiv:1708.00489*, 2017.

Daniel Shanbalico. Dog vs not-dog. https://www.kaggle.com/datasets/danielshanbalico/dog-vs-not-dog. Kaggle dataset, accessed September 2026.

Philip Sun, David Simcha, Dave Dopson, Ruiqi Guo, and Sanjiv Kumar. Soar: improved indexing for approximate nearest neighbor search. *Advances in Neural Information Processing Systems*, 36: 3189–3204, 2023.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Haoru Tan, Sitong Wu, Wei Huang, Shizhen Zhao, and XIAOJUAN QI. Data pruning by information maximization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=93XT0lKOct.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.

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

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, et al. Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Max Welling. Herding dynamical weights to learn. volume 382, pp. 141, 06 2009. doi: 10.1145/1553374.1553517.

Jiayu Wu. Tiny imagenet challenge. 2017. URL https://api.semanticscholar.org/CorpusID:212697711.

Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=7D5EECbOaf9.

Tete Xiao, Yingcheng Liu, Bolei Zhou, Yuning Jiang, and Jian Sun. Unified perceptual parsing for scene understanding. In *European Conference on Computer Vision*, pp. 432–448. Springer, 2018.

Suorong Yang, Peng Ye, Wanli Ouyang, Dongzhan Zhou, and Furao Shen. A clip-powered framework for robust and generalizable data selection. In *International Conference on Learning Representations*, volume 2025, pp. 65723–65741, 2025.

Suorong Yang, Fangjian Su, Hai Gan, Ziqi Ye, Jie Li, Baile Xu, Furao Shen, and Soujanya Poria. Data agent: Learning to select data via end-to-end dynamic optimization. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=fKe6zvEZEE.

Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. Coverage-centric coreset selection for high pruning rates. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=QwKvL6wC8Yi.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127: 302 – 321, 2016. URL https://api.semanticscholar.org/CorpusID:11371972.

Daquan Zhou, Kai Wang, Jianyang Gu, Xiangyu Peng, Dongze Lian, Yifan Zhang, Yang You, and Jiashi Feng. Dataset quantization. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 17159–17170. IEEE, 2023.

Algorithm 1 Online Learning-Signal Propagation

Require: Dataset $\mathcal{D}$, budget ratio r, propagation strength α, recovery rate ρ, neighborhood size k.

Initialize $w_i \leftarrow 1$ for all samples.
for each training epoch **do**
  Reset available candidate set $\mathcal{C}_t \leftarrow \{1, \ldots, N\}$.
  while training budget remains **do**
    Sample mini-batch $B_t \sim P_t$
    Train on B_t and obtain logits o_i and features z_i.
    Compute relative learning signals $\hat{G}_i = G_i/\bar{G}_t$ (Eq. 8), where $G_i = \|p_i - q_i\|_2$ (Eq. 4).
    Refresh cached embeddings $\hat{z}_i \leftarrow z_i/\|z_i\|_2$.
    Retrieve live top-k neighbors for all anchors in B_t:

$$\mathcal{N}_t(B_t) = \text{top-k}\left(\hat{z}_{B_t}\hat{z}_{\mathcal{C}_t}^{\top}\right) .(Eq.\ 9)$$

    Compute all propagated messages in parallel:

$$\ell_{i\to j} = \alpha\, s_{ij}\, \log \hat{G}_i\, \psi_i\, \nu_j .(Eq.\ 10)$$

    Aggregate incoming messages $\Delta_j = |\mathcal{A}_t(j)|^{-1} \sum_{i\in\mathcal{A}_t(j)} \ell_{i\to j}$. (Eq. 11)
    Update priorities $w_j \leftarrow w_j \exp(\Delta_j)$ (Eq. 12) for all reached receivers j.
    Recover priorities: $\tilde{w}_j \leftarrow (1-\rho)w_j + \rho$. (Eq. 16)
  end while
end for

Ensure: Trained model and adaptive sampling priorities w.

A Experiment setup and implementation details

All experiments are conducted on a single NVIDIA H100 GPU. The compute node is equipped with 64 GB of system RAM, 32 CPU cores and 1 TB of local storage. The GPU is used for model training and evaluation, while the CPU resources support data loading, preprocessing and experiment orchestration. All reported results are averaged over three random seeds, with a maximum standard deviation of 0.43 across runs.

Image Classifcation For CIFAR-10 and CIFAR-100, we train ResNet-18 from scratch using SGD with a learning rate of 0.1, weight decay of 5×10^{-4} with batch size 128, following the setup of Tan et al. (2025). Data augmentation consists of RandomResizedCrop and RandomHorizontalFlip. For Tiny-ImageNet, we train a torchvision ResNet-50 from scratch for 90 epochs with batch size 128 and label smoothing of 0.1. We use SGD with weight decay 1×10^{-5} and a OneCycle learning-rate schedule with a maximum learning rate of 0.1. Training images are processed with RandomResizedCrop to 64×64, RandomHorizontalFlip and ImageNet normalization, while validation uses Resize(73), CenterCrop(64) with the same normalization. Performance is evaluated on the full 10,000-image validation set after each epoch.

Table 1 combines results reported in prior work with results reproduced locally under our experimental setup. For CIFAR-10 and CIFAR-100, most static baseline results are taken from the comparison reported by InfoMax (Tan et al., 2025), while Herding (Welling, 2009) is reproduced locally under our setup. Results for all dynamic methods, including Random Dynamic, ϵ-Greedy (Raju et al., 2021), UCB (Raju et al., 2021), InfoBatch (Qin et al., 2024), Data Agent (Yang et al., 2026) and SPADE, are also obtained from our local experiments based on their provided code. All Tiny-ImageNet results are reproduced locally to ensure that the compared methods use the same target-model training configuration.

Semantic Segmentation. We use UPerNet (Xiao et al., 2018) with an ImageNet-pretrained ResNet-50 backbone and a randomly initialized decoder, following the training setup of InfoBatch (Qin et al., 2024). Training combines the main segmentation loss with an auxiliary loss, with candidate-count correction and inverse-budget scaling applied at the image level. Performance is evaluated using validation mean intersection over union (mIoU).

Instruction Tuning. We follow the instruction-tuning setup of InfoBatch (Qin et al., 2024), adapted from prior alpaca instruction-tuning Taori et al. (2023) and dataset quantization (Zhou et al., 2023). Each instruction, response pair is treated as one selectable training example, with the training loss computed only over assistant-response tokens. For selection, we use mean-pooled token embeddings for neighborhood retrieval and derive the per-example learning signal from the logit-gradient magnitude over assistant tokens.

B Further Analysis

B.1 Effect of propagation components and hyperparameters

The propagation mechanism show in Eq. (12).

Removing the sender or receiver replaces the corresponding factor with 1, while geometry remains enabled.

Table 5: Sender/receiver impact on CIFAR-100.

Case	Sender ψ	Receiver ν	Acc. (%)
Full	✓	✓	**77.5**
No sender		✓	75.6
No receiver	✓		76.3
Neither			74.8

Table 5 shows that removing the sender, receiver, or both reduces accuracy by 1.9, 1.2 and 2.7 points, respectively, supporting both factors in this configuration.

Table 6: Effect of top-k on accuracy (%) with a 30% budget and $\alpha = 1$.

k	CIFAR-100	Tiny-ImageNet
64	**77.5**	44.9
128	76.8	**45.9**
256	76.5	45.0
512	76.3	42.0
1024	76.8	41.3
2048	76.7	–

Table 7: Effect of α on Tiny-ImageNet with a 20% budget and $K = 64$.

α	Acc. (%)
1.0	42.9
1.5	**43.1**

The best evaluated k is 64 for CIFAR-100 and 128 for Tiny-ImageNet (Table 6); larger k does not consistently improve accuracy. Increasing α from 1.0 to 1.5 yields a modest gain of 0.2 percentage points (Table 7).

B.2 Does SPADE Benefit from Fixed Pretrained Representations?

SPADE relies on the representation space to identify semantically related candidates for learning-signal propagation. By default, this representation space is produced by the task model itself and is continuously refreshed during training, allowing the neighborhood structure to evolve together with the optimization process. An alternative is to construct the neighborhood space from a fixed pretrained encoder, which may provide stronger semantic structure before the task model has learned sufficiently informative representations.

Table 8: Effect of the representation backbone used for semantic neighborhood propagation on CIFAR-100 and Tiny-ImageNet.

	CIFAR-100					Tiny-ImageNet				
Method	**10%**	**20%**	**30%**	**50%**	**70%**	**10%**	**20%**	**30%**	**50%**	**70%**
SPADE	**71.9**	**76.4**	**77.5**	**78.2**	**79.2**	25.9	**42.9**	44.9	**47.1**	**48.2**
SPADE + DINOv2 (Oquab et al., 2023)	68.2	75.1	76.1	77.1	77.7	27.9	41.8	**45.5**	46.1	47.7
SPADE + SigLIP2 (Tschannen et al., 2025)	66.7	74.0	75.3	76.2	76.5	**31.3**	40.9	44.7	46.8	48.0

To examine this trade-off, we replace SPADE's online-updated embedding cache with fixed representation banks extracted from pretrained encoders. We consider DINOv2 (Oquab et al., 2023) and SigLIP2 (Tschannen et al., 2025) as two representative alternatives. All training examples are embedded once before optimization and these embeddings remain unchanged throughout training. SPADE then applies the same top-k neighbor retrieval, learning-signal propagation and adaptive sampling procedure over the fixed representation space, while the task model itself is trained normally. All other components of SPADE are kept unchanged.

Table 8 shows that fixed pretrained representations do not consistently improve over SPADE's online task-model representation space. On CIFAR-100, the online representation consistently achieves the best performance across all evaluated training budgets. On Tiny-ImageNet, however, fixed pretrained representations can be advantageous under the most restrictive budget: at 10%, DINOv2 improves accuracy from 25.9% to 27.9%, while SigLIP2 further increases it to 31.3%. As the training budget increases, this advantage largely disappears and the online representation becomes competitive or superior in most settings.

These results suggest that the usefulness of the representation source depends on the stage and amount of task-specific learning available to the model. Pretrained encoders can provide a useful semantic prior when only a small fraction of the training data is observed, particularly when the task model has not yet developed a sufficiently discriminative representation space. In contrast, with greater training exposure, the task model can progressively adapt its features to the target objective, making the resulting neighborhoods more aligned with the learning signals that SPADE propagates. Overall, the results favor online task-adaptive representations as the default design, while also indicating that pretrained representations can be beneficial in low-budget scenarios.

B.3 Potential acceleration with incremental ANN indexing

The current implementation computes similarities between every query embedding in a mini-batch and the complete embedding bank. For batch size B, dataset size N and embedding dimension d, the per-step retrieval cost is

$$C_{\text{exact}} = O(BNd). \tag{19}$$

This exhaustive search returns exact nearest neighbors, but its cost grows linearly with the dataset size. A natural alternative is approximate nearest-neighbor (ANN) search using indexing structures such as HNSW(Malkov & Yashunin, 2020) or ScaNN(Guo et al., 2020; Sun et al., 2023). These methods trade a small reduction in neighbor recall for substantially lower retrieval cost. Recent ScaNN implementations further support online and online-incremental updates, allowing embeddings to be refreshed without rebuilding the complete index.

For a partitioned ANN index with L leaves and q searched leaves per query, assuming approximately balanced partitions, the search cost can be approximated as

$$C_{\text{search}} \approx O\left[Bd\left(L + q\frac{N}{L}\right)\right]. \tag{20}$$

Using the common heuristic $L \approx \sqrt{N}$ gives

$$C_{\text{search}} \approx O\left(B(q+1)\sqrt{N}d\right), \tag{21}$$

Table 9: Estimated computational reduction of incremental partitioned ANN search relative to exhaustive neighborhood search. The estimate assumes $q = 10$ searched leaves and includes the cost of incremental embedding updates.

Dataset size N	$\sqrt{N}$	**Estimated reduction**
10^6	1,000	$83.3\times$
10^7	3,162	$263.5\times$
10^8	10,000	$833.3\times$
10^9	31,623	$2{,}635.2\times$

compared with $O(BNd)$ for exhaustive retrieval.

In our lazy-refresh setting, only the B embeddings observed in the current mini-batch require index updates. Let $U(N, d)$ denote the amortized cost of one incremental update. The total per-step ANN cost becomes

$$C_{\mathrm{ANN}} \approx O\left(B(q+1)\sqrt{N}d\right) + B\,U(N,d). \tag{22}$$

Hence, the expected acceleration over exhaustive search is

$$S_{\mathrm{total}} = \frac{BNd}{B(q+1)\sqrt{N}d + B\,U(N,d)}. \tag{23}$$

To illustrate the potential gain, assume that an incremental update requires approximately one partition-assignment operation, $U(N, d) \approx \sqrt{N}d$. Then,

$$S_{\mathrm{total}} \approx \frac{\sqrt{N}}{q+2}. \tag{24}$$

To illustrate the potential scalability, we set $q = 10$ searched leaves and assume an incremental update cost of approximately one partition-assignment operation, giving

$$S_{\mathrm{total}} \approx \frac{\sqrt{N}}{q+2} = \frac{\sqrt{N}}{12}. \tag{25}$$

The potential benefit therefore grows with dataset size: under the above assumptions, the estimated comparison reduction increases from approximately $83\times$ at one million samples to more than $2{,}600\times$ at one billion samples. These values represent theoretical reductions in neighborhood-search computation rather than measured wall-clock speedups.

C Proof

C.1 Per-example decomposition of the learning signal

Although optimization is performed on a batch loss, standard supervised cross-entropy is additive across examples. For a batch $B_t = \{(x_i, y_i)\}_{i=1}^{m}$, we write

$$L_{B_t} = \frac{1}{m} \sum_{i \in B_t} L_i, \tag{26}$$

where the per-example cross-entropy loss is

$$L_i = -\sum_{k=1}^{C} q_i[k] \log p_i[k], \tag{27}$$

with logits $o_i \in \mathbb{R}^C$ and softmax probabilities

$$p_i[k] = \frac{\exp(o_i[k])}{\sum_{\ell=1}^{C} \exp(o_i[\ell])}. \tag{28}$$

Using the log-softmax identity,

$$\log p_i[k] = o_i[k] - \log\left(\sum_{\ell=1}^{C} \exp(o_i[\ell])\right), \tag{29}$$

the derivative with respect to logit $o_i[r]$ is

$$\frac{\partial \log p_i[k]}{\partial o_i[r]} = \frac{\partial o_i[k]}{\partial o_i[r]} - \frac{\exp(o_i[r])}{\sum_{\ell=1}^{C} \exp(o_i[\ell])} \tag{30}$$

$$= \delta_{kr} - p_i[r], \tag{31}$$

where δ_{kr} denotes the Kronecker delta. Therefore,

$$\frac{\partial L_i}{\partial o_i[r]} = -\sum_{k=1}^{C} q_i[k] \frac{\partial \log p_i[k]}{\partial o_i[r]} \tag{32}$$

$$= -\sum_{k=1}^{C} q_i[k] \left(\delta_{kr} - p_i[r]\right) \tag{33}$$

$$= -q_i[r] + p_i[r] \sum_{k=1}^{C} q_i[k]. \tag{34}$$

Since q_i is a probability distribution, $\sum_{k=1}^{C} q_i[k] = 1$, giving

$$\frac{\partial L_i}{\partial o_i[r]} = p_i[r] - q_i[r]. \tag{35}$$

Collecting all output coordinates,

$$\frac{\partial L_i}{\partial o_i} = p_i - q_i. \tag{36}$$

This quantity is defined independently for every example before batch aggregation. Indeed, by linearity of differentiation,

$$\nabla_\theta L_{B_t} = \nabla_\theta \left(\frac{1}{m} \sum_{i \in B_t} L_i\right) \tag{37}$$

$$= \frac{1}{m} \sum_{i \in B_t} \nabla_\theta L_i. \tag{38}$$

Let

$$J_i = \frac{\partial o_i}{\partial \theta} \tag{39}$$

denote the Jacobian of the logits with respect to the model parameters. Applying the chain rule,

$$\nabla_\theta L_i = J_i^\top \frac{\partial L_i}{\partial o_i} = J_i^\top (p_i - q_i), \tag{40}$$

and hence

$$\nabla_\theta L_{B_t} = \frac{1}{m} \sum_{i \in B_t} J_i^\top (p_i - q_i). \tag{41}$$

Equation 41 shows that the batch gradient is formed by aggregating example-specific learning signals. We therefore define

$$G_i = \|p_i - q_i\|_2 = \left\| \frac{\partial L_i}{\partial o_i} \right\|_2, \tag{42}$$

which measures the magnitude of the optimization signal induced by example i at the model output. Importantly, G_i is available from the standard forward-pass outputs and does not require constructing a separate full parameter gradient for each training example.

D Extended Related Work

Static data selection. Static data-selection methods construct a reduced training set using signals derived from training dynamics, optimization behavior, or representation-space structure. Forgetting Events (Toneva et al., 2018) scores examples by correct-to-incorrect prediction transitions during training, while GraNd and EL2N (Paul et al., 2021) use gradient-norm and prediction-error statistics, respectively, to identify important examples early in optimization. Geometry-based approaches instead emphasize representativeness and coverage. Herding (Welling, 2009) greedily selects examples to approximate feature-space statistics, while K-Center (Sener & Savarese, 2017) seeks a subset that geometrically covers the remaining data. Moderate Coreset (Xia et al., 2023) selects samples whose importance scores lie near the score median, using distance to the class center as its proposed scoring criterion. CCS (Zheng et al., 2023) jointly considers sample importance and data coverage, improving robustness particularly at high pruning rates. More recent approaches explicitly incorporate relationships between examples. D^2-Pruning (Maharana et al., 2023) propagates difficulty scores over a dataset graph and uses the resulting scores to balance difficult and diverse regions, whereas InfoMax (Tan et al., 2025) combines sample-wise informativeness with pairwise similarity to favor informative but non-redundant coresets. In the coreset setting considered here, these methods ultimately produce a subset that remains fixed during subsequent target-model training.

Dynamic data selection. Dynamic methods adapt data priorities as model training progresses. Online Batch Selection (Loshchilov & Hutter, 2015) ranks examples by their latest observed loss and assigns higher sampling probability to higher-ranked examples. Dynamic Data Pruning (Raju et al., 2021) revises the selected training data throughout optimization using random, ϵ-Greedy, and UCB-based strategies. InfoBatch (Qin et al., 2024) dynamically prunes a portion of lower-loss examples according to the loss distribution and rescales retained samples to compensate for the resulting change in gradient contribution. Data Agent (Yang et al., 2026) instead formulates selection as a sequential decision problem and learns a sample-wise PPO policy guided by loss-based difficulty and confidence-based uncertainty signals. These methods adapt to changes in sample utility during training, but their importance estimates are primarily updated from statistics of directly evaluated samples, or require an additional learned selection policy.

Online neighborhood-aware exposure. SPADE combines the temporal adaptivity of dynamic selection with the inter-sample structure used by graph- and similarity-based pruning. D^2-Pruning (Maharana et al., 2023) and InfoMax (Tan et al., 2025) use relationships between examples to construct an informative coreset. In contrast, SPADE uses every observed mini-batch as fresh feedback for future sampling: learning signals from the current examples are propagated through their representation-space neighborhoods to candidates that have not yet been observed at the current model state. The

propagated signals update persistent sampling priorities at mini-batch granularity, continuously reshaping the distribution over the remaining candidates rather than producing a fixed subset. This allows informative regions to receive greater training exposure while maintaining support for lower-priority examples, without training a separate selection policy.