BWS: Best Window Selection Based on Sample Scores for Data Pruning across Broad Ranges

Hoyong Choi [*1] **Nohyun Ki** [*2] **Hye Won Chung** [2]

Abstract

Data subset selection aims to find a smaller yet informative subset of a large dataset that can approximate the full-dataset training, addressing challenges associated with training neural networks on large-scale datasets. However, existing methods tend to specialize in either high or low selection ratio regimes, lacking a universal approach that consistently achieves competitive performance across a broad range of selection ratios. We introduce a universal and efficient data subset selection method, Best Window Selection (BWS), by proposing a method to choose the best window subset from samples ordered based on their difficulty scores. This approach offers flexibility by allowing the choice of window intervals that span from easy to difficult samples. Furthermore, we provide an efficient mechanism for selecting the best window subset by evaluating its quality using kernel ridge regression. Our experimental results demonstrate the superior performance of BWS compared to other baselines across a broad range of selection ratios over datasets, including CIFAR-10/100 and ImageNet, and the scenarios involving training from random initialization or fine-tuning of pre-trained models.

1. Introduction

In many machine learning tasks, the effectiveness of deep neural networks often relies on large-scale datasets that include a vast number of samples, enabling them to achieve state-of-the-art performances. However, working with such large datasets presents several challenges, including the high computational costs, storage requirements, and potential concerns related to privacy (Schwartz et al., 2020; Strubell et al., 2019). Data subset selection emerges as a promising approach to address these issues. This involves the careful selection of a smaller, yet highly informative, subset from the original large dataset. The goal is to find a subset with a specified selection ratio that approximates the performance of the entire dataset or incurs minimal performance loss.

Data subset selection has two primary approaches: score-based selection and optimization-based selection. Score-based selection involves defining a specific score to measure each sample's influence (Koh & Liang, 2017), difficulty (Toneva et al., 2019; Paul et al., 2021), or consistency (Jiang et al., 2021) in training neural networks. The primary goal is to identify the most valuable or influential samples within the dataset while pruning the remaining samples that have minimal impact on the model's generalization ability. On the other hand, optimization-based selection approaches find the optimal subset of a fixed size that can best approximate the full dataset training in terms of loss gradient or curvature by solving the associated optimization problem (Mirzasoleiman et al., 2020; Pooladzandi et al., 2022; Shin et al., 2023; Yang et al., 2023). The original optimization, which is NP-hard, is commonly approximated by submodular functions and a greedy algorithm is adopted to sequentially select the samples up to the size limit of the subset.

While the prior approaches successfully reduce dataset size in specific scenarios, there is not a single selection method that universally outperforms other baselines across broad selection ratios. To illustrate this, we conduct a benchmark comparison between two methods: Forgetting score (Toneva et al., 2019) representing the score-based selection approach, and LCMat (Shin et al., 2023) representing the optimization-based selection approach. We evaluate the test accuracy of models trained with different subset sizes of datasets, including CIFAR-10/100 (Krizhevsky, 2009) and ImageNet (Deng et al., 2009), ranging from 1% to 90%, as selected by these two methods (Table 1). Score-based methods, which prioritize samples of high influence or difficulty, tend to initially select rare yet influential samples while excluding typical or easy samples. These methods demonstrate competitive performance, nearly matching the full-dataset training, when the selection ratio is sufficiently

[*]Equal contribution [1]Samsung Research, Seoul, South Korea [2]School of Electronic Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea. Correspondence to: Hye Won Chung <hwchung@kaist.ac.kr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

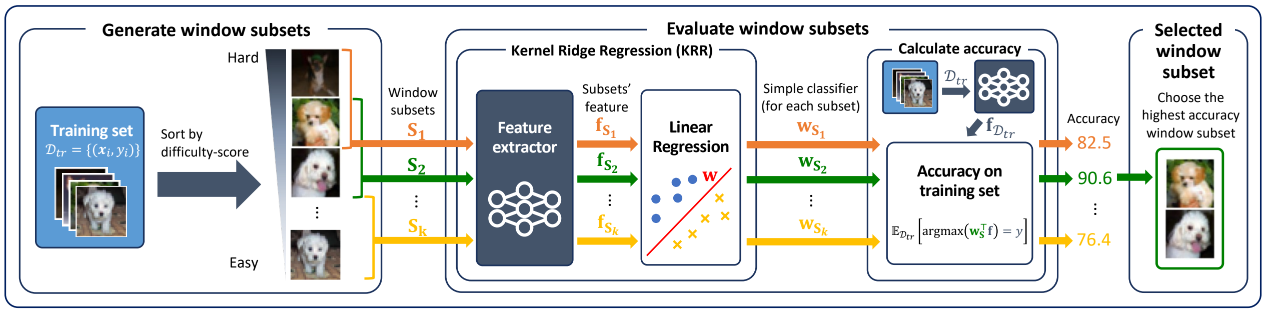


Figure 1. Overview of the proposed method, Best Window Selection (BWS). BWS is composed of two parts, 1) generating window subsets and 2) evaluating window subsets. We first sort samples by a difficulty score (e.g., Forgetting (Toneva et al., 2019)) and generate window subsets of a fixed size while varying their starting points. We then evaluate the window subsets, by solving kernel ridge regression on the input features of each window subset and obtaining simple (linear) classifiers associated with each window subset. Finally, we evaluate the performance of these classifiers on the full training dataset to identify the best window subset achieving the highest accuracy.

high (e.g., over 40% for CIFAR-10). However, they suffer significant performance degradation as the selection ratio decreases. In contrast, optimization-based methods tend to select representative samples that best approximate the full dataset training. Consequently, they achieve competitive performance even with very low selection ratios. However, their performance gains are limited as the selection ratio increases due to lack of diversity in sample selection. These findings show the variability in the criteria for an effective data subset, depending on the selection ratio, and highlight that previous methods may not be general enough to cover the entire spectrum of selection ratios.

Our key contribution is the development of a universal and efficient data selection method capable of maintaining competitive performance across a wide range of selection ratios. We introduce the Best Window Selection (BWS) method, illustrated in Fig. 1. The key idea involves ordering samples based on their difficulty-based sample scores and offering flexibility in choosing a 'window subset' from the ordered samples. Here, the window subset is defined as a subset consisting of samples with a contiguous ranking of difficulty. By allowing the starting point (the ranking of the hardest data in the subset) of each window subset to vary, we enable the selection of easy, moderate, or hard data subsets. We first demonstrate the existence of the best window that achieves the highest test accuracy for each subset size, and reveal that the optimal starting point for the best window varies depending on both the subset size and dataset. We then present a computationally-efficient method for selecting the best window subset without the need to evaluate models trained with each subset. We achieve this by solving a kernel ridge regression problem using samples from each window, evaluating the corresponding solution's performance on the full training dataset, and selecting the best performing window subset.

We evaluate our selection method, BWS on CIFAR-10/100 and ImageNet, demonstrating that BWS consistently outperforms other baselines, including both score-based and optimization-based approaches, across a wide range of selection ratios ranging from 1% to 90%. For CIFAR-10, BWS achieves a 15-30% improvement in test accuracy compared to Forgetting (Toneva et al., 2019) in the low selection ratios of 1-10%. It also demonstrates competitive performance in the high selection ratio regime, reaching up to 93% test accuracy with only a 40% data subset. BWS also consistently outperforms optimization-based techniques such as LCMat (Shin et al., 2023) and AdaCore (Pooladzandi et al., 2022), despite requiring significantly lower computational costs. Furthermore, we empirically verify that BWS is effective across different model architectures, including pre-trained ViT (Dosovitskiy et al., 2021). Another significant advantage of our method is its resilience to label noise, enhancing its robustness in sample selection. Our code is publicly available at https://github.com/NohyunKi/BWS.

2. Related Works

Score-based selection Some initial works in score-based selection use validation/test sets to quantify the effect of each training instance. Data Shapley (Ghorbani & Zou, 2019; Kwon et al., 2021; Kwon & Zou, 2022) evaluates the value of each instance by measuring the average change in validation accuracy when that instance is excluded from the dataset. Influence Function (Koh & Liang, 2017; Pruthi et al., 2020) approximates how a model's prediction changes as individual training examples are visited. In the absence of a validation set, score-based selection quantifies the learning difficulty or consistency of samples during neural network training. Forgetting (Toneva et al., 2019) and EL2N (Paul et al., 2021) introduce a difficulty score to measure a data point's learning difficulty. Memorization (Feldman & Zhang, 2020) and C-score (Jiang et al., 2021) aim to predict the accuracy on a sample when the full dataset is utilized, except for that sample. CG-score (Ki et al., 2023) evaluates data instances without model training by calculating the analytical gap in generalization errors when an instance is held out. These score-based methods prioritize difficult or influential samples for data subset selection. While they effectively select a subset approximating the full-dataset performance, their performance degrades significantly as the selection ratio decreases, as achieving high performance solely with difficult samples becomes challenging.

Optimization-based selection Optimization-based selection involves formulating an optimization problem to select a coreset of a given size that can effectively approximate the diverse characteristics of the full dataset. These methods include coreset selection to approximate the training distribution by herding (Chen et al., 2010) or k-center algorithms (Sener & Savarese, 2018). Recent approaches have sought subsets of samples approximating loss gradients or curvature by CRAIG (Mirzasoleiman et al., 2020), CREST (Yang et al., 2023), and AdaCore (Pooladzandi et al., 2022). While these methods have proven effective, they are computationally demanding and necessitate full-dataset sampling at each epoch. LCMat (Shin et al., 2023) addresses this computational challenge by aligning both gradients and Hessians without requiring periodic full-dataset sampling. However, these methods often struggle to choose diverse samples, and their performance does not match that of score-based approaches, in the intermediate to high selection ratio regimes.

In contrast to these approaches, we develop a universal selection method capable of consistently identifying a high-performance subset across a wide range of selection ratios. While recent methods like Moderate-DS (Xia et al., 2023) and CCS (Zheng et al., 2023) have also aimed for universality across various selection ratios, our method outperforms these approaches, over a broad range of selection ratios, as demonstrated in Section 6. Moderate-DS selects samples closest to the median of the features of each class, while CCS prunes a β% of hard examples, with β being a hyperparameter, and then selects samples with a uniform difficulty score distribution. Importantly, our method does not require hyperparameter tuning, such as setting β in CCS, since we assess the quality of window subsets and efficiently find the best one using kernel ridge regression.

3. Motivation

3.1. No single method prevails over the entire range

We conduct an evaluation of existing data selection methods across a wide range of selection ratios. Specifically, we benchmark two representative methods: Forgetting score (Toneva et al., 2019), representing difficulty score-based selection, and LCMat (Shin et al., 2023), representing optimization-based selection. We assess the test accuracy of

Table 1. Test accuracy across various selection ratios for the CIFAR-10/100 and ImageNet datasets, with subsets selected using random sampling, Forgetting score (Toneva et al., 2019), and LCMat (Shin et al., 2023). The best performance among the three is highlighted in **bold**.

Selection ratio		1%	5%	10%	20%	30%	40%	50%	75%	90%	100%
CIFAR-10	Random	39.10	**67.14**	**78.43**	86.87	89.91	91.66	92.83	94.40	95.08	95.40
	Forgetting	30.08	42.39	54.31	79.19	89.13	**93.41**	**94.49**	**95.31**	**95.14**	
	LCMat	**41.53**	66.86	77.48	**87.34**	**90.72**	92.45	93.38	94.90	95.19	
CIFAR-100	Random	5.89	23.76	42.03	55.03	**65.98**	**69.23**	**73.84**	76.53	78.29	78.81
	Forgetting	7.01	20.69	34.22	50.95	61.54	68.92	72.65	**78.55**	**79.69**	
	LCMat	**8.43**	**28.51**	**42.81**	**55.77**	64.39	67.22	73.11	77.51	78.47	
ImageNet	Random	**6.14**	**33.17**	45.87	59.19	65.94	68.23	70.14	73.74	74.83	75.85
	Forgetting	4.78	28.18	45.84	**60.75**	**67.48**	**70.26**	**72.73**	**74.63**	**75.53**	
	LCMat	6.01	32.26	**46.08**	59.02	65.28	68.50	70.30	74.13	74.81	

models trained on subsets of CIFAR-10/100 and ImageNet, with selection ratios ranging from 1% to 90%, as summarized in Table 1. For the Forgetting score approach, we sort the samples in descending order based on their scores, defined as the number of times during training the decision of that sample switches from a correct one to incorrect one, and select the top-ranking (most difficult) samples. In contrast, for LCMat, we employ an optimization to identify a subset that best approximates the loss curvature of the full dataset. We employ ResNet18 (He et al., 2016) for CIFAR-10 and ResNet50 for CIFAR-100 and ImageNet.

We can observe that the most effective strategy varies depending on the selection ratios, and there is no single method that consistently outperforms others across the entire range of selection ratios. Specifically, for CIFAR-10 with low subset ratios (1-30%), the optimization-based selection (LCMat) performs better than the difficulty score-based selection (Forgetting). In this regime, the 'Forgetting' even underperforms random selection. However, as the subset ratio increases beyond 40%, the 'Forgetting' outperforms both the LCMat and random selection. Similar trends are observed for CIFAR-100 and ImageNet. Interestingly, for CIFAR-100, there is an intermediate regime where neither the 'Forgetting' nor LCMat outperform random sampling.

These findings emphasize that the desired properties of data subsets change depending on the selection ratios. In cases of low selection ratios (sample-deficient regime), it is more beneficial to identify a representative subset that closely resembles the full dataset in terms of average loss gradients or curvature during training. However, as the selection ratio increases (sample-sufficient regime), preserving the high-scoring, rare or difficult-to-learn samples becomes more critical, as these samples are known to enhance the generalization capability of neural networks and cannot be fully captured by a representative subset that reflects only the average behavior of the full dataset.

3.2. Theoretical analysis

To validate this experimental finding, we provide a theoretical analysis of optimal subset selection, revealing similar change of trends in the desirable subsets depending on the selection ratios. We consider a binary classification problem by solving a linear regression problem, as detailed below: Data samples $\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n \in \mathbb{R}^d$ are generated from a multivariate normal distribution, $\mathcal{D} = \frac{1}{\sqrt{d}}\mathcal{N}(0, \mathbf{I}_d)$. The label y_i of sample $\mathbf{x}_i$ is determined by the sign of its first element. Specifically, if $(\mathbf{x}_i)_1 > 0$ then $y_i = 1$; and if $(\mathbf{x}_i)_1 < 0$, then $y_i = -1$. We define the score of each sample as $1/|(\mathbf{x}_i)_1|$. Samples closer to the decision boundary $(\mathbf{x})_1 = 0$ have higher scores, while those farther from the boundary have lower scores. We select a label-balanced subset of size m, denoted by $(\mathbf{X_S}, \mathbf{y_S}) \in \mathbb{R}^{d\times m} \times \{-1, 1\}^m$, and use it to solve a linear regression problem to find $\mathbf{w_S} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{y_S} - \mathbf{X_S}^\top \mathbf{w}\|_2^2$. For a new sample $\mathbf{x}'$, our decision will be $+1$ if $\mathbf{w_S}^\top \mathbf{x}' > 0$ and -1 otherwise. Thus, we consider $\mathbf{w_S}$ to be a better solution when the value of its first element, $(\mathbf{w_S})_1$, is larger. For the above setup, we analyze the solution $\mathbf{w_S}$ depending on the subset size $|\mathbf{S}|$.

A similar problem setup was analyzed in (Sorscher et al., 2022), demonstrating that the optimal selection strategy varies depending on the subset ratio. Specifically, Sorscher et al. (2022) considers a max margin classifier trained on a data subset selected by the teacher-perceptron model, providing a comprehensive set of equations enabling numerical computation of the generalization error for various subset data distributions. In contrast, our contribution lies in providing a closed-form solution for the optimal linear classifier, as summarized in the theorem below. This theorem shows the transition of the optimal sample selection strategy between sample-deficient and sample-sufficient regimes.

Theorem 1 (Informal)**.** *If the subset size is as small as* $|\mathbf{S}| = m \ll \sqrt{d/\ln d}$*, then the first coordinate of* $\mathbf{w_S}$ *is approximated as* $(\mathbf{w_S})_1 \approx \sum_{i=1}^m |(\mathbf{x}_i)_1|$*. On the other hand, if* $|\mathbf{S}| = m \gg d^2 \ln d$*, it can be approximated as* $(\mathbf{w_S})_1 \approx (\sum_{i=1}^m |(\mathbf{x}_i)_1|)/(\sum_{i=1}^m |(\mathbf{x}_i)_1|^2)$.

A more formal statement and the proof of Thm. 1 is available in Appendix A.2. From Thm.1, it is evident that the characteristics of the desirable data subset $\mathbf{X_S}$ vary depending on the subset size regime. In the sample-deficient regime ($m \ll \sqrt{d/\ln d}$), it is more advantageous to include samples that are farther from the decision boundary (easy samples) in $\mathbf{X_S}$ to train a better classifier, resulting in a higher value of $(\mathbf{w_S})_1$. Conversely, in the sample-sufficient regime ($m \gg d^2 \ln d$), it is more beneficial to include samples closer to the decision boundary (difficult samples) to increase $(\mathbf{w_S})_1$. We conjecture that the relatively wide gap between two distinct regimes ($[\sqrt{d/\ln d}, d^2 \ln d]$) may be attributed to the loose analysis. We anticipate that a more precise boundary will occur at $m = \Theta(d)$, where $m \ll d$ ($m \gg d$) corresponds to the sample-deficient (sufficient) regime. We provide empirical results that support this theoretical analysis and our conjecture in Appendix A.3.

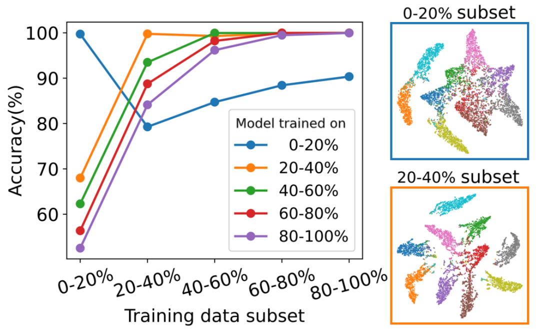


Figure 2. Results on "training set split" experiment on CIFAR-10 dataset, when five different models are trained by five different data subsets, divided by their difficulty rankings, $[0, 20]\%$ (hardest) to $[80, 100]\%$ (easiest). Model accuracies (y-axis) are evaluated on all five subsets (x-axis) separately. Right figures visualize the t-SNE of test samples' features extracted from models trained by the hardest $[0, 20]\%$ subset (top) and the $[20, 40]\%$ subset (bottom).

Having identified the distinct properties of desirable data subsets depending on the subset size, the remaining question is how to design a universal data selection method capable of performing well across a wide range of selection ratios.

4. Window Subset: Flexible Subset Selection

4.1. Desirable difficult level for data subsets in moderate selection ratios? Hard, but not the hardest

The underlying rationale for difficulty score-based selection methods like Forgetting (Toneva et al., 2019) and EL2N (Paul et al., 2021) is that training models on a subset consisting of challenging data will enable the models to learn (or memorize) the atypical features of hard samples, while still retaining the capacity to learn typical features of easier samples. However, as shown in our empirical findings in Sec. 3.1 and further supported by our theoretical analysis in Sec. 3.2, this assumption may not hold when the subset ratio is extremely small. This leads us to our next question: Is it still feasible for models trained on *hard* instances to effectively learn *easier* instances, without having been exposed to these samples during training, at *moderate* selection ratios?

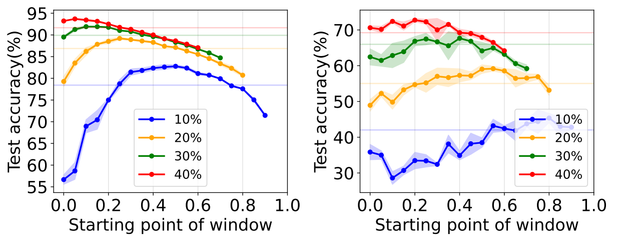
To investigate this, we design a "training set split" experiment on CIFAR-10 dataset. We divide the training dataset into five subsets and observe the impact of training on each subset on the accuracy across the other subsets. In detail, we sort the CIFAR-10 training instances by forgetting score (Toneva et al., 2019) and divide them into five subsets based on consecutive ranking intervals: the hardest 20% (rankings within $[0, 20]\%$), $[20, 40]\%$, and so on, up to the easiest 20% ($[80, 100]\%$). We train five different ResNet18 models, each on one of these subsets, and then evaluate their classification accuracies on all five subsets separately.

The results, presented in Fig. 2, reveal that models trained on harder data subsets generally perform better across all subsets, with the exception of the model trained solely on the hardest 20%. For instance, the model trained on the $[20, 40]\%$-ranked subset effectively classifies instances not only within its training range but also those in the easier $[40, 100]\%$ range. This suggests that training with harder instances helps the model learn both the unique features of these challenging instances and the common, representative features of the entire dataset. This finding supports the rationale behind existing score-based selection methods, which prioritize selecting challenging data for training.

Yet, this pattern does not hold for the model trained exclusively on the *hardest* 20% subset. This model exhibits a significant drop in accuracy across all the easier subsets, except for the hardest subset it was trained on. This indicates that a model trained with only the most challenging instances lacks generalizability to easier samples.

We support this claim by analyzing the feature spaces of models trained with the hardest $[0, 20]\%$ subset and the subsequent $[20, 40]\%$ subset. Our focus is on demonstrating that the model trained with the hardest 20% subset struggles to effectively create a feature space for classification. We extract features of CIFAR-10 test samples from each model and visualize their t-SNE (van der Maaten & Hinton, 2008) in Fig. 2 (right). The figure reveals that the feature space generated by the model trained on the hardest subset does not efficiently cluster test samples by class. We further quantify this using the *neural collapse* property (Kothapalli, 2023), which compares within-class feature variability to inter-class feature variability. Let $f_{k,i}$ be the feature of the i-th data in the k-th class, $\mu_k = \frac{1}{n}\sum_{i=1}^{n} f_{k,i}$ be the feature mean of class k, and $\mu_G = \frac{1}{K}\sum_{k=1}^{K} \mu_k$ be the global mean feature. The within-class covariance Σ_w is defined by $\frac{1}{Kn}\sum_{k=1}^{K}\sum_{i=1}^{n}(f_{k,i} - \mu_k)(f_{k,i} - \mu_k)^\top$, and the inter-class covariance Σ_B by $\frac{1}{K}\sum_{k=1}^{K}(\mu_k - \mu_G)(\mu_k - \mu_G)^\top$. The trace, $\text{tr}(\Sigma_W \Sigma_B^\dagger)$, then measures the clusterability of features with respect to their classes, with a lower value indicating better clustering. The $\text{tr}(\Sigma_W \Sigma_B^\dagger)$ values for models trained on $[0, 20]\%$ (the hardest 20%), $[20, 40]\%$, and so on, up to $[80, 100]\%$, and the full dataset, are 9.33, 1.68, 1.99, 2.60, 3.35, and 1.04, respectively. Notably, there is a significant increase in $\text{tr}(\Sigma_W \Sigma_B^\dagger)$ for the hardest subset, suggesting poor feature learning for classification.

In summary, training with harder data generally benefits learning both representative and atypical features, aiding in better model generalization. However, when the subset ratio is moderate and the subset consists of the hardest samples, the model may suffer significant performance drop and fail to establish an effective feature learning for classification.



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

Figure 3. Sliding window experiments to measure the test accuracy of the models trained by window subsets while changing the starting point of the windows in CIFAR-10 (left) and CIFAR-100 (right) dataset. Samples are sorted in descending order by their difficulty scores. The horizontal lines are results from random selection. For each subset ratio, there exists the best window, and its starting point shifts toward left as the subset ratio increases. Results for ImageNet dataset is also reported in Appendix F.1

4.2. The existence of a high-performing window subset

Section 4.1 implies that for each subset ratio, there is a proper difficulty level of the subset for better model generalization. Expecting that a subset composed of samples of proper difficult level will perform well, we consider the window selection method, similar to (Lee & Chung, 2024), that selects a window subset from samples ordered by their difficulty scores. In detail, we sort the samples in descending order based on their difficulty scores and select a starting point, such as $s\%$ for a given window size of $w\%$, to choose continuous intervals of samples within $[s, s + w]\%$. This approach has two merits: 1) flexibility and 2) computational-efficiency. The flexibility in choosing the starting point $s\%$ of the window allows us to opt for easy, moderate, or hard data subsets depending on the choice of the starting point. The search space of window selection method is confined to the number of possible starting points for the windows, making the window selection method computationally much more efficient compared to a general subset selection where the search space scales as $\binom{n}{m} \approx \exp(cn)$ for some constant $c > 0$ when the subset size m is a constant fraction of n.

We first explore the performance of the window selection approach while varying the starting point and illustrate the existence of the best window subset. We sort the samples from CIFAR-10/100 in descending order based on their Forgetting scores (Toneva et al., 2019), and select windows of different sizes, ranging from 10% to 40%, by adjusting the starting point from 0 to $(100 - w)\%$ with a step size of 5%. We then train ResNet18 for CIFAR-10 and ResNet50 for CIFAR-100 using the windows subsets and plot the resulting test accuracies in Fig. 3.

We can observe that, for each subset ratio, there exists an optimal starting point, and this optimal point shifts towards

Algorithm 1 BWS: Best Window Selection Method

```
Input Dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ sorted by difficulty scores from the hardest to easiest, subset size $m$, and step size $t$.
  Train a feature extractor $f(\cdot)$ by $m$ randomly chosen samples from the dataset.
  Extract the features of the samples by using $f(\cdot)$ and denote them by $\mathbf{f}_i = [f(\mathbf{x}_i), 1]$.
  for $k \in \{0, t, 2t, 3t \dots, \lfloor (n-m)/t \rfloor t\}$ do
    Define a window subset $\mathbf{S} = \{(\mathbf{f}_i, y_i)\}_{i=k}^{k+m-1}$.
    for $c \in \{1, 2, \dots C\}$ do
      For the samples in $\mathbf{S}$ with label $c$, set the label equal to 1. For others, set the label to 0.
      Solve the linear regression problem Eq.1 with the window subset $\mathbf{S}$. Let $\mathbf{w}_\mathbf{S}(c)$ be the solution.
    end for
    Obtain $\mathbf{w}_\mathbf{S} \in \mathbb{R}^{(d+1)\times C}$ by $\mathbf{w}_\mathbf{S} := [\mathbf{w}_\mathbf{S}(1), \dots \mathbf{w}_\mathbf{S}(C)]$.
    Calculate the accuracy of $\mathbf{w}_\mathbf{S}$ by
    $\frac{1}{n}\sum_{i=1}^n \mathbb{1}(\arg\max_c(\mathbf{w}_\mathbf{S}^\top \mathbf{f}_i)_c = y_i)$.
  end for
Output Window subset $\mathbf{S}$ for which the accuracy of $\mathbf{w}_\mathbf{S}$ is maximized.
```

lower values (indicating more difficult samples) as the window subset size increases. Specifically, for CIFAR-10, the optimal window subset of size 10% falls within the interval $[50, 60]\%$, while for a window size of 40%, it falls within $[5, 45]\%$. Similar trends are observed for CIFAR-100, albeit with distinct optimal starting points depending on the dataset. For CIFAR-100, with a window size of 10%, the best window subset comprises samples from $[80, 90]\%$, primarily consisting of easy samples. It is important to note that the 10% subset for CIFAR-100 includes only 50 samples per class, whereas for CIFAR-10, it includes 500 samples per class. Consequently, the optimal 10% window for CIFAR-100 ($[80, 90]\%$) tends to include more easy and representative samples capable of capturing the representative features of each class.

The observation that the optimal starting point of the window subset varies based on both the subset size and the dataset introduces a new challenge in window selection: How can we efficiently identify the best window subset without having to evaluate models trained on each subset? We address this crucial question by introducing a proxy task to estimate the quality of window subsets.

5. Best Window Selection (BWS)

Our goal is to develop a computationally-efficient method capable of assessing and identifying the best window subset without requiring the training of a model on every potential subset. To achieve this goal, we propose to solve a kernel ridge regression (KRR) problem by using each window subset and evaluate the performance of the corresponding

Table 2. Comparison between the window subsets chosen by the sliding window experiment in Fig. 3 (left) and BWS (using the KRR as a proxy task) (right) in terms of their starting points and test accuracies. The chosen windows align well between the two.

Ratio	Sliding window experiment		BWS	
	Starting point	Test accuracy	Starting point	Test accuracy
10%	50%	82.67	55%	82.29
20%	25%	89.06	30%	88.74
30%	15%	91.80	15%	91.80
40%	5%	93.59	5%	93.59

solution on the full training datasets. Using KRR for a proxy task is motivated by the observation that the kernel regression with the model-related kernels can provide a good approximation to the original model (Neal, 1996; Lee et al., 2018; Jacot et al., 2018; Arora et al., 2019), while providing computational efficiency compared to training the actual models. We provide further justifications of this proxy task in Appx. B. Alg. 1 outlines the main steps.

Let $\mathbf{f}_i := [f(\mathbf{x}_i), 1] \in \mathbb{R}^{d+1}$ be the feature vector of $\mathbf{x}_i$ obtained by a feature extractor $f(\cdot)$. The details of the feature extractor is available in the end of this section. For each window subset $\mathbf{S} = \{(\mathbf{f}_i, y_i)\}_{i=1}^m$ composed of m samples, define $\mathbf{X}_\mathbf{S} := [\mathbf{f}_1, \dots, \mathbf{f}_m]$ and $\mathbf{y}_\mathbf{S} := [y_1, \dots, y_m]$. Then, we denote the problem of kernel ridge regression, and the corresponding solution, using the subset $\mathbf{S}$ by

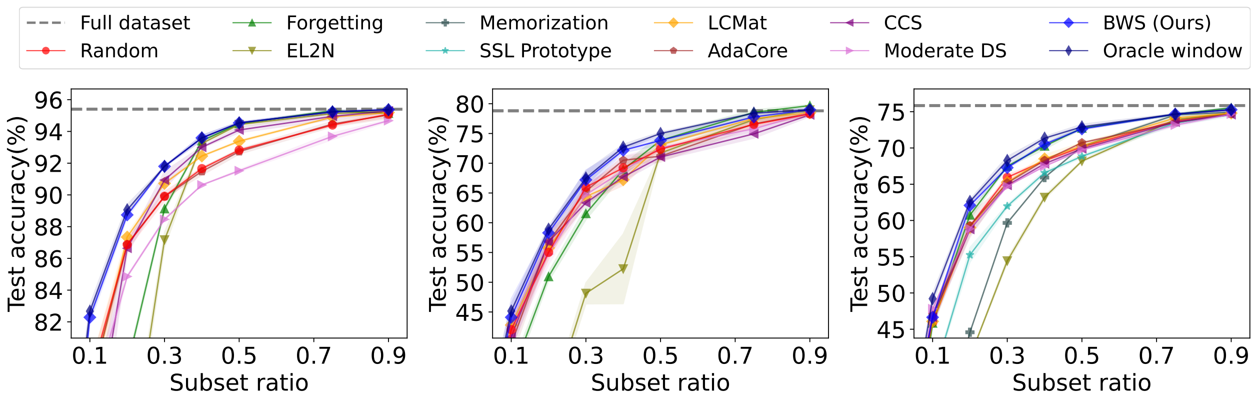
$$\mathbf{w}_\mathbf{S} := \arg\min_{\mathbf{w}} \|\mathbf{y}_\mathbf{S} - \mathbf{X}_\mathbf{S}^\top \mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2, \quad (1)$$

$$\begin{aligned}\mathbf{w}_\mathbf{S} &= (\lambda \mathbf{I}_{d+1} + \mathbf{X}_\mathbf{S}{\mathbf{X}_\mathbf{S}}^\top)^{-1}\mathbf{X}_\mathbf{S}\mathbf{y}_\mathbf{S} \\ &= \mathbf{X}_\mathbf{S}(\lambda \mathbf{I}_m + \mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1}\mathbf{y}_\mathbf{S}. \quad (2)\end{aligned}$$

We set $\lambda = 1$ to prevent singularity in matrix inversion. The matrix inversion in Eq. 2 can be performed efficiently in a lower dimension between $d+1$ and m.

Our algorithm finds the best window subset by evaluating the performance of $\mathbf{w}_\mathbf{S}$, corresponding to each window subset $\mathbf{S}$, on classifying the training samples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ as described in Alg. 1. To apply $\mathbf{w}_\mathbf{S}$ for C-class classification problem, we find $\mathbf{w}_\mathbf{S}(c) \in \mathbb{R}^{d+1}$ for each class $c \in \{1, \dots, C\}$, classifying whether a sample belongs to class c or not, and simply place the vectors in columns of $\mathbf{w}_\mathbf{S} \in \mathbb{R}^{(d+1)\times C}$. Then, we evaluate the performance of $\mathbf{w}_\mathbf{S}$ by calculating the classification accuracy $\frac{1}{n}\sum_{i=1}^n \mathbb{1}(\arg\max_c(\mathbf{w}_\mathbf{S}^\top \mathbf{f}_i)_c = y_i)$ on the full training set.

In Table 2, we compare the performances of window subsets chosen by the sliding window experiment in Fig. 3 and BWS (using the KRR as a proxy task) on CIFAR-10 dataset. We compare the starting points and test accuracies of the window subsets chosen by the two different methods for each subset ratio. We can observe that window subsets chosen by KRR align well with those chosen by the sliding



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

Figure 4. (a, b, c) **Data pruning experiments.** Test accuracy of the models trained with data subsets of varying ratios in CIFAR-10/100, and ImageNet dataset, selected by different methods. Our method (BWS) outperforms other baselines across a wide range of selection ratios and achieves the accuracy as high as the Oracle window. Full results are reported in Table 13–15.

window experiment. This observation demonstrates the effectiveness of our algorithm, which can efficiently replace the need to train models on each window subset and evaluate them on test dataset. BWS also finds the near optimal starting points for CIFAR-100 and ImageNet datasets across a broad range of subset ratios (from 1% to 90%). The detailed results are available in the Appendix H.

Feature extractor When $|\mathbf{S}| = m$, we randomly select m samples from the full dataset, and use these samples to train a neural network for a few epochs to generate a feature extractor $f(\cdot)$. For CIFAR-10, we train ResNet18 for 20 epochs, and for CIFAR-100/ImageNet, we train ResNet50 for 20 epochs. The rationale behind training a feature extractor with random samples matching the window subset size is to simulate the situation where the model is trained with a limited window subset of the same size, enabling effective quality evaluation for window subsets.

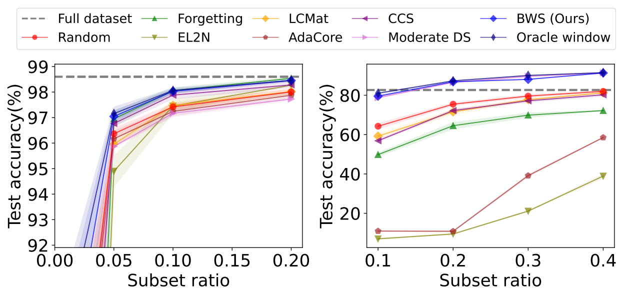
Computational complexity The computational complexity of Algorithm 1 includes training a feature extractor and solving the regression problem for $(\lfloor (n-m)/t \rfloor)$-subsets. Training the feature extractor is relatively efficient since it involves only a few epochs. Solving the regression requires matrix inversion, which takes $O(\min(d,m)^3)$ steps, with $d = 512$ for ResNet18 and 2048 for ResNet50. This cost is significantly lower than other optimization-based baselines. For example, running BWS for the CIFAR-10 dataset with ResNet-18 and a step size of 5% takes less than 11 seconds. Detailed comparisons are provided in Appendix C.3.

6. Experiments

To demonstrate the effectiveness of our method, we conduct data pruning experiments. We select a subset of the dataset using each selection method while pruning the rest of the samples, and evaluate the performance of the model trained with each subset. We perform these experiments using ResNet18 for CIFAR-10 and ResNet50 for CIFAR-100 and ImageNet. Baselines include 1) two difficulty score-based selection: Forgetting and EL2N, 2) two optimization-based selection: AdaCore and LCMat, and 3) two universal selection methods: Moderate DS score and CCS. We also add SSL Prototype (Sorscher et al., 2022) and memorization score (Feldman & Zhang, 2020) as baselines on the ImageNet experiment, since these scores are known to achieve competitive performances especially on the large-scale datasets (Sorscher et al., 2022). More details about the baselines and experiments are available in Appx. C. The full experimental results are available in Appx. H.

6.1. Experimental Results

Data pruning experiments In Fig. 4, we present the test accuracies of models trained with data subsets of varying ratios, selected by different methods. The reported values are mean, and the shaded regions are std. across three (two) independent runs for CIFAR-10/100 (ImageNet). The Oracle window curve represents the results obtained using the window subset of the highest test accuracy found by the sliding window experiment as in Fig. 3, and BWS represents the results obtained using Alg. 1. We can observe that our method, BWS, consistently outperforms all other baselines across almost all selection ratios, and achieves the performance near the Oracle window. In the case of CIFAR-10/100, the difficulty score-based methods, Forgetting and EL2N, perform well in high ratio regimes but experience significant performance drop as the selection ratio decreases. The optimization-based methods, LCMat and AdaCore, achieve better performance than the difficulty score-based methods



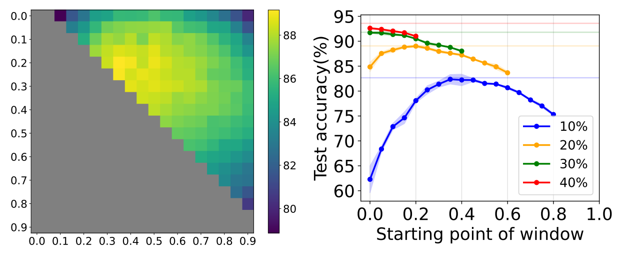
(a) CIFAR-10 with ViT (b) Noisy CIFAR-10

Figure 5. (a) **Cross architecture experiment.** Test accuracy of the model fine-tuned with subsets of varying ratios in the CIFAR-10 dataset, selected by different methods. We utilize the Vision Transformer (ViT) architecture, pretrained on the ImageNet dataset. (b) **Robustness to label noise.** Data pruning experiments with CIFAR-10, including 20% label-noise. For both experiments, BWS surpasses other baselines for a wide range of selection ratios.

in low selection ratios but underperform in high selection ratios. Detailed numbers are reported in Table 13–15.

Cross-architecture experiments To test the robustness of our method across changes in model architectures, we conduct data pruning experiments on CIFAR-10 while using different architectures during sample scoring and training. Window subsets are constructed using samples ordered by their Forgetting scores, calculated on ResNet18, and then the best window selection (Alg. 1) and the model training are conducted using a simpler CNN/EfficientNet-B0 or a larger Vision Transformer (ViT) (Dosovitskiy et al., 2021), pre-trained on the ImageNet. The results on the ViT are presented in Fig. 5(a), while those on CNN and EfficientNet-B0 are shown in Fig. 9 of Appx. F.2. In all cases, our method consistently achieves competitive performances across all selection ratios, demonstrating its robustness to changes in neural network architectures during data subset selection.

Robustness to label noise Additionally, we demonstrate that BWS is robust against label noise in subset selection. Existing sample selection methods, which rely on difficulty-based sample scores (Toneva et al., 2019; Paul et al., 2021), are susceptible to a particular limitation: they often assign high scores to samples corrupted by label noise, as these samples are inherently hard to learn. This poses the risk of unintentionally selecting noisy samples during the selection phase. On the contrary, our BWS algorithm adopts a different approach by solving a proxy task using kernel ridge regression rather than solely relying on high or low difficulty-based scores. We test the robustness of BWS in the presence of label noise by corrupting randomly chosen 20% samples of CIFAR-10 dataset by random label noise. To further enhance the robustness of our method, we modify Alg. 1 to evaluate the solution of kernel ridge regression using only the low-scoring 50% samples from the training dataset, which will rarely include label-noise samples, instead of the full dataset. In Fig. 5(b), we compare the performance of this modified version of BWS with other baselines. While difficulty score-based selection and optimization-based selection methods suffer from performance degradation due to label noise, our method, along with another label noise-robust method, Moderate DS, achieves performance even higher than what is achievable with the full training dataset, which includes the 20% label noise. Further experiment results with higher noise ratio are provided in Appendix F.3.



(a) Two half-width windows (b) Wider window

Figure 6. (a) Test accuracies of the models trained with two half-width windows of varying starting points. Each axis indicates the starting points of each widow, and brighter color indicates higher accuracy. The best result is observed near the diagonal, contiguous windows. (b) Test accuracy of the models trained with wider windows. Horizontal lines are the results of oracle window subset. At high ratio, oracle window outperforms wider windows.

6.2. Ablation study

BWS operates by sorting training instances based on their difficulty scores, creating window subsets, and selecting the best window by a proxy task. To assess the importance of each component, we conduct several ablation studies.

Different types of window subsets Our method employs a window type that includes samples from a contiguous range of difficulty scores while changing the starting point. We explore two more generalized window types: a union of two half-width windows and a wider window where the samples are randomly selected from a wider range. For two half-width windows, given a subset of size w, we search over all combinations of two half-width windows, denoted by $[x_1, x_1+w/2] \cup [x_2, x_2+w/2]$, while varying their starting points $x_1 \in [0, 100-w]$ and $x_2 \in [x_1+w/2, 100-w/2]$ with a step size of 5%. For wider windows, we consider a window that is c times wider than the subset size w, denoted as $[x_1, x_1+c \cdot w]$ while varying the starting point x_1 within the range $[0, 100-c \cdot w]$ with a step size of 5%. These ablation studies are conducted with CIFAR-10 on ResNet18, to see whether the generality in subset selection can bring

Table 3. The maximum test accuracy achieved by each window type in CIFAR-10. The best contiguous window nearly matches two half-width windows and outperforms wider windows.

Selection Ratio	Two half-width windows	Twice wider window	Best contiguous window
10%	83.04	82.37	82.67
20%	89.16	89.01	89.06
30%	92.02	91.72	91.80
40%	93.67	92.62	93.59

Table 4. Test accuracy of the models trained by window subsets of CIFAR-10 selected by different proxy tasks. Our method achieves the better performance, and the best window subsets selected by ours aligns better with those of oracle windows.

Proxy task	Subset ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%
SVP	Test accuracy	46.25	71.35	80.95	88.06	90.68	91.63	93.36	94.75	95.37
	Window index	80%	65%	60%	40%	30%	25%	15%	5%	0%
Gradient difference	Test accuracy	39.45	70.40	82.24	88.42	90.68	91.63	92.72	94.30	94.82
	Window index	50%	45%	40%	35%	30%	25%	20%	10%	5%
Gradient similarity	Test accuracy	36.33	60.46	74.77	87.79	91.77	93.59	94.54	95.23	95.37
	Window index	30%	25%	20%	15%	10%	5%	0%	0%	0%
BWS	Test accuracy	46.10	70.70	82.29	88.74	91.80	93.59	94.54	95.23	95.37
	Window index	90%	70%	55%	30%	15%	5%	0%	0%	0%
Oracle	Test accuracy	47.17	72.89	82.67	89.06	91.80	93.59	94.54	95.23	95.37
	Window index	85%	55%	50%	25%	15%	5%	0%	0%	0%

meaningful gain possibly at the cost of computation.

In Table 3, we present the maximum test accuracies achieved by the two non-contiguous (two half-width/wider windows) and contiguous window types. Remind that two half-width window type includes all the contiguous windows. We observe that for every subset ratio, the performance of the best contiguous window subset almost matches that of two half-width windows, and outperforms wider widows. Moreover, Fig. 6(a) shows that the best composition of two half-width windows occur when the two windows are close to each other (the diagonal positions in the figure). The sliding window experiment for wider windows in Fig. 6(b) shows that the best contiguous window (horizontal lines) achieves better performance than wider windows, especially in high ratios. These results support our use of contiguous window subsets in choosing the near-optimal subset in a computationally efficient manner across a broad range of selection ratios. Further results are reported in Appx. G.1–G.2.

Different types of proxy task We also evaluate the effectiveness of our proxy task, kernel ridge regression in Alg. 1, by comparing it with three different variants: 1) Selection via proxy (SVP) (Coleman et al., 2020), utilizing a smaller model (ConvNet) for choosing the best window, 2) Gradient ℓ_2-norm difference, which finds a window subset minimizing the ℓ_2-norm difference between the average gradients of the full dataset and the window subset, and 2) Gradient cosine similarity, which finds a window subset maximizing the cosine similarity between the average gradients of the full dataset and the window subset. The last two methods are inspired by gradient-matching strategies used in optimization-based coreset selection (Mirzasoleiman et al., 2020; Yang et al., 2023). Table 4 presents the test accuracies achieved by models trained on window subsets selected by each method, along with the corresponding starting points of the chosen windows. The last row shows the result with the oracle window. Our method achieves better test accuracy compared to the three variants, and the window selected by our method aligns better with the oracle selection. In particular, SVP tends to select easier subsets possibly due to the limited capacity of the simple network used in the proxy task. This result demonstrates that the best subset cannot be effectively chosen by using a simpler network or matching the average gradients; it requires a proxy task such as kernel ridge regression, with model-related kernels, to evaluate the quality of window subsets for classification tasks. We also perform an ablation study to show the robustness of our method across various difficulty scores in Appx. G.3.

7. Conclusion and Discussion

We introduced the Best Window Selection (BWS), a universal and efficient data subset selection method capable of achieving competitive performance across a wide range of selection ratios. Our experimental results demonstrate that BWS effectively identifies the best window subset from samples ordered by difficulty-based scores, utilizing a simple proxy task based on kernel ridge regression. This method outperforms previous data subset selection approaches, which often excel within a limited range of selection ratios.

Subset selection has become a crucial technique in the big data era, allowing for the reduction of large datasets with minimal information loss. However, current efforts, including BWS, mainly focus on sample selection for supervised learning on curated datasets designed for classification tasks with well-defined labels. The next stage for subset selection may involve addressing challenges associated with much larger and more complex datasets. For instance, DataComp (Gadre et al., 2023) proposes a new benchmark for subset selection by providing a web-scale multimodal dataset as the full training set. This setup challenges researchers to develop strategies for selecting subsets that benefit diverse downstream test sets capable of zero-shot generalization.

We believe that the insights gained through BWS–specifically, the shifts in the desired dataset characteristics based on selection ratio and the methodology for efficiently identifying the optimal subset using a simple proxy task–may provide valuable perspectives for designing data filtering or selection strategies for these large-scale datasets.

Impact Statement

This paper addresses the performance degradation seen in existing data subset selection methods when the selection ratio varies widely. We introduce a methodology specifically designed to effectively counter this challenge. Our proposed universal data subset selection method delivers consistent, competitive performance across various selection ratios. This is particularly valuable in practical situations where computational and storage resources for training can vary, necessitating flexible sample selection based on the required subset ratios.

Acknowledgements

This research was supported by the National Research Foundation of Korea under grant 2021R1C1C11008539.

References

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R., and Wang, R. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, 2019.

Chen, Y., Welling, M., and Smola, A. Super-samples from kernel herding. In *Conference on Uncertainty in Artificial Intelligence*, 2010.

Citovsky, G., DeSalvo, G., Kumar, S., Ramalingam, S., Rostamizadeh, A., and Wang, Y. Leveraging importance weights in subset selection. In *International Conference on Learning Representations*, 2023.

Coleman, C., Yeh, C., Mussmann, S., Mirzasoleiman, B., Bailis, P., Liang, P., Leskovec, J., and Zaharia, M. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2020.

Culotta, A. and McCallum, A. Reducing labeling effort for structured prediction tasks. In *Association for the Advancement of Artificial Intelligence*, 2005.

Das, S., Singh, A., Chatterjee, S., Bhattacharya, S., and Bhattacharya, S. Finding high-value training data subset through differentiable convex programming. In *Machine Learning and Knowledge Discovery in Database*, 2021.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition*, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

Fan, Z. and Wang, Z. Spectra of the conjugate kernel and neural tangent kernel for linear-width neural networks. In *Advances in Neural Information Processing Systems*, 2020.

Feldman, V. and Zhang, C. What neural networks memorize and why: Discovering the long tail via influence estimation. In *Advances in Neural Information Processing Systems*, 2020.

Gadre, S. Y., Ilharco, G., Fang, A., Hayase, J., Smyrnis, G., Nguyen, T., Marten, R., Wortsman, M., Ghosh, D., Zhang, J., Orgad, E., Entezari, R., Daras, G., Pratt, S., Ramanujan, V., Bitton, Y., Marathe, K., Mussmann, S., Vencu, R., Cherti, M., Krishna, R., Koh, P. W., Saukh, O., Ratner, A., Song, S., Hajishirzi, H., Farhadi, A., Beaumont, R., Oh, S., Dimakis, A., Jitsev, J., Carmon, Y., Shankar, V., and Schmidt, L. Datacomp: In search of the next generation of multimodal datasets, 2023.

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

Har-Peled, S., Roth, D., and Zimak, D. Maximum margin coresets for active and noise tolerant learning. In *International Joint Conference on Artificial Intelligence*, 2007.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, 2018.

Jiang, Z., Zhang, C., Talwar, K., and Mozer, M. C. Characterizing structural regularities of labeled data in overparameterized models. In *International Conference on Machine Learning*, 2021.

Just, H. A., Kang, F., Wang, T., Zeng, Y., Ko, M., Jin, M., and Jia, R. LAVA: Data valuation without pre-specified learning algorithms. In *International Conference on Learning Representations*, 2023.

Ki, N., Choi, H., and Chung, H. W. Data valuation without training of a model. In *International Conference on Learning Representations*, 2023.

Killamsetty, K., Sivasubramanian, D., Ramakrishnan, G., De, A., and Iyer, R. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, 2021a.

Killamsetty, K., Sivasubramanian, D., Ramakrishnan, G., and Iyer, R. Glister: Generalization based data subset selection for efficient and robust learning. In *Association for the Advancement of Artificial Intelligence*, 2021b.

Koh, P. W. and Liang, P. Understanding black-box predictions via influence functions. In *Advances in Neural Information Processing Systems*, 2017.

Kong, S. T., Jeon, S., Na, D., Lee, J., Lee, H.-S., and Jung, K.-H. A neural pre-conditioning active learning algorithm to reduce label complexity. In *Advances in Neural Information Processing Systems*, 2022.

Kothapalli, V. Neural collapse: A review on modelling principles and generalization. In *Transactions on Machine Learning Research*, 2023.

Krizhevsky, A. Learning multiple layers of features from tiny images. Technical report, 2009.

Kwon, Y. and Zou, J. Beta shapley: a unified and noise-reduced data valuation framework for machine learning. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Kwon, Y., A. Rivas, M., and Zou, J. Efficient computation and analysis of distributional shapley values. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Lee, J., Bahri, Y., Novak, R., Schoenholz, S. S., Pennington, J., and Sohl-Dickstein, J. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Lee, Y. and Chung, H. W. SelMatch: Effectively scaling up dataset distillation via selection-based initialization and partial updates by trajectory matching. In *International Conference on Machine Learning*, 2024.

Maini, P., Garg, S., Lipton, Z. C., and Kolter, J. Z. Characterizing datapoints via second-split forgetting. In *Advances in Neural Information Processing Systems*, 2022.

Mirzasoleiman, B., Bilmes, J., and Leskovec, J. Coresets for data-efficient training of machine learning models. In *Proceedings of the 37th International Conference on Machine Learning*, 2020.

Neal, R. M. *Bayesian learning for neural networks*. Springer Science & Business Media, 1996.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, 2019.

Paul, M., Ganguli, S., and Dziugaite, G. K. Deep learning on a data diet: Finding important examples early in training. In *Advances in Neural Information Processing Systems*, 2021.

Pleiss, G., Zhang, T., Elenberg, E. R., and Weinberger, K. Q. Identifying mislabeled data using the area under the margin ranking. In *Advances in Neural Information Processing Systems*, 2020.

Pooladzandi, O., Davini, D., and Mirzasoleiman, B. Adaptive second order coresets for data-efficient machine learning. In *International Conference on Machine Learning*, 2022.

Pruthi, G., Liu, F., Kale, S., and Sundararajan, M. Estimating training data influence by tracing gradient descent. In *Advances in Neural Information Processing Systems*, 2020.

Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O. Green ai. In *Communications of the ACM*, 2020.

Sener, O. and Savarese, S. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

Shin, S., Bae, H., Shin, D., Joo, W., and Moon, I.-C. Loss-curvature matching for dataset selection and condensation. In *International Conference on Artificial Intelligence and Statistics*, 2023.

Sorscher, B., Geirhos, R., Shekhar, S., Ganguli, S., and Morcos, A. S. Beyond neural scaling laws: beating power law scaling via data pruning. In *Advances in Neural Information Processing Systems*, 2022.

Strubell, E., Ganesh, A., and McCallum, A. Energy and policy considerations for deep learning in nlp. In *Association for Computational Linguistics*, 2019.

Swayamdipta, S., Schwartz, R., Lourie, N., Wang, Y., Hajishirzi, H., Smith, N. A., and Choi, Y. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Conference on Empirical Methods in Natural Language Processing*, 2020.

Tan, M. and Le, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, 2019.

Toneva, M., Sordoni, A., des Combes, R. T., Trischler, A., Bengio, Y., and Gordon, G. J. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

van der Maaten, L. and Hinton, G. Visualizing data using t-SNE. In *Journal of Machine Learning Research*, 2008.

Wu, Z., Shu, Y., and Low, B. K. H. DAVINZ: Data valuation using deep neural networks at initialization. In *International Conference on Machine Learning*, 2022.

Xia, X., Liu, J., Yu, J., Shen, X., Han, B., and Liu, T. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *International Conference on Learning Representations*, 2023.

Yang, Y., Kang, H., and Mirzasoleiman, B. Towards sustainable learning: Coresets for data-efficient deep learning. In *International Conference on Machine Learning*, 2023.

Zheng, H., Liu, R., Lai, F., and Prakash, A. Coverage-centric coreset selection for high pruning rates. In *International Conference on Learning Representations*, 2023.

Zhou, Y., Nezhadarya, E., and Ba, J. Dataset distillation using neural feature regression. In *Advances in Neural Information Processing Systems*, 2022.

A. Proof of Theoretical Analysis

A.1. Linear ridge regression

The solution of the linear ridge regression problem is derived as follows.

$$L(\mathbf{w}) = \|\mathbf{y} - \mathbf{X}^\top \mathbf{w}\|_2^2 + \lambda \|\mathbf{w}\|_2^2$$
$$\frac{\partial L}{\partial \mathbf{w}} = 2\mathbf{X}\mathbf{X}^\top \mathbf{w} - 2\mathbf{X}\mathbf{y} + 2\lambda \mathbf{w} = 0 \quad \Rightarrow \quad \mathbf{w} = (\lambda \mathbf{I} + \mathbf{X}\mathbf{X}^\top)^{-1}\mathbf{X}\mathbf{y}$$
$$\therefore \mathbf{w}_\mathbf{S} = (\lambda \mathbf{I} + \mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)^{-1}\mathbf{X}_\mathbf{S}\mathbf{y}_\mathbf{S} = \mathbf{X}_\mathbf{S}(\lambda \mathbf{I} + \mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1}\mathbf{y}_\mathbf{S}$$

A.2. Proof of Theorem 1

In this section, we provide the detailed proof of Theorem 1 in Sec. 3.2. We assume that $n = poly(d)$ data inputs $\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_n$ are sampled from normalized multivariate normal distribution, $\mathcal{D} = \frac{1}{\sqrt{d}}\mathcal{N}(0, \mathbf{I}_d) = \frac{1}{\sqrt{d}}(\mathcal{N}_1, \mathcal{N}_2 \dots \mathcal{N}_d)$ where $\{\mathcal{N}_k\}$ are $i.i.d.$ normal distributions. Remind that the label y_i of sample $\mathbf{x}_i$ is determined by the sign of its first element, i.e., if $(\mathbf{x}_i)_1 > 0$ then $y_i = 1$, and if $(\mathbf{x}_i)_1 < 0$, then $y_i = -1$. We select a subset of size m, denoted by $(\mathbf{X}_\mathbf{S}, \mathbf{y}_\mathbf{S}) \in \mathbb{R}^{d\times m} \times \{-1, 1\}^m$.

We first provide a high-level proof idea of Theorem 1. Note that the optimal $\mathbf{w}_\mathbf{S} = \arg\min_{\mathbf{w}\in\mathbb{R}^d} \|\mathbf{y}_\mathbf{S} - \mathbf{X}_\mathbf{S}^\top \mathbf{w}\|_2^2$ can be written as $\mathbf{w}_\mathbf{S} = \mathbf{X}_\mathbf{S}(\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1}\mathbf{y}_\mathbf{S}$ when $m \le d$, and $\mathbf{w}_\mathbf{S} = (\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)^{-1}\mathbf{X}_\mathbf{S}\mathbf{y}_\mathbf{S}$ when $m \ge d$. Let us first consider the case of $m \le d$. Due to the properties of high dimensional multivariate normals, we have $\|\mathbf{x}_i\| \in [1 \pm \sqrt{7\ln n/2d}]$ for all $i \in [n]$ and $|\mathbf{x}_i^\top \mathbf{x}_j| \le \sqrt{7\ln n/2d}$ for all $i \ne j \in [n]$ with high probability. Thus, $\|\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S} - \mathbf{I}_m\|_F \le \sqrt{\frac{m^2(7\ln n)}{2d}}$ where $\mathbf{I}_m$ is the identity matrix of size m. When $m \ll \sqrt{d/\ln d}$, we have $(\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S}) \approx (\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1} \approx \mathbf{I}_m$, and thus $\mathbf{w}_\mathbf{S} = \mathbf{X}_\mathbf{S}(\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1}\mathbf{y}_\mathbf{S} \approx \mathbf{X}_\mathbf{S}\mathbf{y}_\mathbf{S}$, which implies that $(\mathbf{w}_\mathbf{S})_1 \approx \sum_{i=1}^m |(\mathbf{x}_i)_1|$. Let us next consider the case of $m \ge d$. Note that the diagonal terms of $\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top$ are $\sum_{i=1}^m |(\mathbf{x}_i)_k|^2 = \Theta(m/d)$ for $k \in [d]$ and the off-diagonal terms are $\sum_{i=1}^m (\mathbf{x}_i)_k(\mathbf{x}_i)_l = O(\sqrt{m\ln d}/d)$ for $k \ne l \in [d]$ with high probability. The eigenvalues of $\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top$ can be shifted from its diagonal entries $(\sum_{i=1}^m |(\mathbf{x}_i)_1|^2, \dots, \sum_{i=1}^m |(\mathbf{x}_i)_d|^2)$ by at most $\frac{\sqrt{m\ln d}}{d} d = \sqrt{m\ln d}$ by the effect of its off-diagonal entries. Thus, when $m/d \gg \sqrt{m\ln d}$, i.e., $m \gg d^2\ln d$, we can have $\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top \approx \mathrm{diag}(\sum_{i=1}^m |(\mathrm{x_i})_1|^2, \dots, \sum_{i=1}^m |(\mathrm{x_i})_d|^2)$ and $(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)^{-1} \approx \mathrm{diag}((\sum_{i=1}^m |(\mathrm{x_i})_1|^2)^{-1}, \dots, (\sum_{i=1}^m |(\mathrm{x_i})_d|^2)^{-1})$. Since $\mathbf{w}_\mathbf{S} = (\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)^{-1}\mathbf{X}_\mathbf{S}\mathbf{y}_\mathbf{S}$, the first coordinate value of $\mathbf{w}_\mathbf{S}$ is $(\mathbf{w}_\mathbf{S})_1 \approx (\sum_{i=1}^m |(\mathbf{x}_i)_1|)/(\sum_{i=1}^m |(\mathbf{x}_i)_1|^2)$.

To more formally state and prove Theorem 1, we provide Theorem 2 to explain the regime of low selection ratio ($m = o(\sqrt{d/\ln d})$) and Theorem 3 for the high selection ratio ($m = \omega(d^2\ln d)$). To prove the two theorems, we use the following three lemmas, including the tail bounds on chi-square and Gaussian distributions, and Gershgorin theorem, which are stated as below:

Lemma A.1 (Chi-square tail bound). *If* $\mathbf{x} \sim \chi^2(d)$*, then* $\mathbb{P}(\chi^2(d) \ge d + 2\sqrt{dt} + 2t) \le e^{-t}$ *and* $\mathbb{P}(\chi^2(d) \le d - 2\sqrt{dt}) \le e^{-t}$.

Lemma A.2 (Gaussian tail bound). *If* $\mathbf{x} \sim \mathcal{N}(0, 1)$*, then* $\mathbb{P}(|\mathbf{x}| \ge t) \le e^{\frac{-t^2}{2}}$.

Lemma A.3 (Gershgorin circle theorem). *Let* $A \in \mathbb{C}^{d\times d}$ *be a matrix with its* (i, j)*-th entry equal to* a_{ij}*. Let* $r_i := \sum_{j\ne i} |a_{ij}|$ *and* $D_i := D_{r_i}(a_{ii})$ *be a closed ball centered* a_{ii} *with radius* r_i*. Then, every eigenvalue of* A *is contained in* $\cup_i D_i$

Gershgorin circle theorem restricts the eigenvalues of a matrix in a union of disks, whose centers are diagonal elements, and the radius is the sum of off-diagonal elements.

Now, we provide Theorem 2, which will be used to explain why selecting low-scoring (easy) data samples results in a good performance when the subset size $|\mathbf{S}|$ is small.

Theorem 2 (Sample-deficient regime). *If* $m = o\left(\sqrt{d/\ln d}\right)$*, then* $\|(\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1} - \mathbf{I}_m\|_2 \le m\sqrt{\frac{7\ln n}{2d}}$ *with high probability as* $d \to \infty$.

Proof. At first, we prove two properties of the high dimensional multivariate normal distribution, which state that the norm of every $\mathbf{x}_i$ is almost equal to 1, and every two independent vectors are almost orthogonal for large enough d. For any

$1 \leq i \neq j \leq n$, with probability $1 - O(\frac{1}{n})$, we have

$$1 - \sqrt{\frac{7 \ln n}{2d}} \leq \|\mathbf{x}_i\|_2 \leq 1 + \sqrt{\frac{7 \ln n}{2d}}, \quad \text{and} \tag{3}$$

$$|\mathbf{x}_i^\top \mathbf{x}_j| < \sqrt{\frac{7 \ln n}{2d}}. \tag{4}$$

The first property (Eq. 3) can be proved by Lemma A.1. Let $t = 3 \ln n$ for Lemma A.1. Then,

$$\mathcal{P}(\chi^2(d) \geq d + 2\sqrt{3d \ln n} + 6 \ln n) \leq \frac{1}{n^3} \quad \text{and} \quad \mathcal{P}(\chi^2(d) \leq d - 2\sqrt{3d \ln n}) \leq \frac{1}{n^3}.$$

Since $2\sqrt{3d \ln n} + 6 \ln n \leq \sqrt{13 d \ln n}$ for large enough d, with probability $1 - O(\frac{1}{n^3})$ we have

$$1 - \sqrt{\frac{13 \ln n}{d}} \leq \frac{1}{d}\chi^2(d) \leq 1 + \sqrt{\frac{13 \ln n}{d}} \xrightarrow{\text{d} \to \infty} 1 - \sqrt{\frac{7 \ln n}{2d}} \leq \sqrt{\frac{1}{d}\chi^2(d)} \leq 1 + \sqrt{\frac{7 \ln n}{2d}}. \tag{5}$$

Since $\mathbf{x}_i \sim \frac{1}{\sqrt{d}}\mathcal{N}(0, \mathbf{I}_d)$ and $\|\mathbf{x}_i\|_2^2 = \frac{1}{d}\chi^2(d)$, for $\forall i \in [n]$, with probability $1 - O(\frac{1}{n^2})$, Eq. 3 follows.

The proof of the second property (Eq. 4) also utilizes Lemma A.1. Let $\mathbf{x}_i = \dfrac{1}{\sqrt{d}}(\mathcal{N}_{i1}, \mathcal{N}_{i2}, \ldots \mathcal{N}_{id})$ and $\mathbf{x}_j = \dfrac{1}{\sqrt{d}}(\mathcal{N}_{j1}, \mathcal{N}_{j2}, \ldots \mathcal{N}_{jd})$ where $\mathcal{N}_{ik}, \mathcal{N}_{jk}$ are $i.i.d.$ normals $\mathcal{N}(0, 1)$. Then,

$$\begin{aligned}
\mathbf{x}_i^\top \mathbf{x}_j &= \frac{1}{d}\sum_{k=1}^{d} \mathcal{N}_{ik}\mathcal{N}_{jk} = \frac{1}{d}\sum_{k=1}^{d} \frac{(\mathcal{N}_{ik} + \mathcal{N}_{jk})^2 - (\mathcal{N}_{ik} - \mathcal{N}_{jk})^2}{4} \\
&= \frac{1}{2d}\sum_{k=1}^{d} \left[\left(\frac{\mathcal{N}_{ik} + \mathcal{N}_{jk}}{\sqrt{2}}\right)^2 - \left(\frac{\mathcal{N}_{ik} - \mathcal{N}_{jk}}{\sqrt{2}}\right)^2\right] = \frac{1}{2d}\sum_{k=1}^{d}[(\mathcal{N}_k^{'})^2 - (\mathcal{N}_k^{''})^2] \\
&= \frac{1}{2d}(\chi_1^2(d) - \chi_2^2(d)),
\end{aligned}$$

where $\mathcal{N}_k^{'}$ and $\mathcal{N}_k^{''}$ are $i.i.d.$ normals, and $\chi_1^2(d)$ and $\chi_2^2(d)$ are $i.i.d$ chi-squares.

As shown in Eq. 5, with probability $1 - O(\frac{1}{n^3})$,

$$1 - \sqrt{\frac{7 \ln n}{2d}} \leq \sqrt{\frac{1}{d}\chi_1^2(d)} \quad \text{and} \quad \sqrt{\frac{1}{d}\chi_2^2(d)} \leq 1 + \sqrt{\frac{7 \ln n}{2d}}. \tag{6}$$

Thus, we have

$$\left|\frac{1}{2d}(\chi_1^2(d) - \chi_2^2(d))\right| \leq \sqrt{\frac{7 \ln n}{2d}}. \tag{7}$$

By applying a union bound, for $\forall i \neq j \in [n]$, with probability $1 - O(\frac{1}{n})$, we have $|\mathbf{x}_i^\top \mathbf{x}_j| \leq \sqrt{\frac{7 \ln n}{2d}}$. From Eq. 3 and Eq. 4, we obtain that $\|\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S} - \mathbf{I}_m\|_F^2 \leq m^2 \left(\frac{7 \ln n}{2d}\right)$.

Let $\mathbf{A} = \mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S}$, then we derive the bound on $\|\mathbf{I} - \mathbf{A}^{-1}\|_2$ from the bounds of $\|\mathbf{I} - \mathbf{A}\|_2$ and $\|\mathbf{A}^{-1}\|_2$. First, note that

$$\|\mathbf{I} - \mathbf{A}\|_2 \leq \|\mathbf{I} - \mathbf{A}\|_F \leq m\sqrt{\frac{7 \ln n}{2d}} \quad \text{and} \quad m\sqrt{\frac{7 \ln n}{2d}} \to 0 \text{ as } d \to \infty$$

since $m = o(\sqrt{d/\ln d})$ and $n = poly(d)$. Moreover, we have

$$\begin{aligned}
\|\mathbf{A}^{-1}\|_2 &= \|(\mathbf{I} - (\mathbf{I} - \mathbf{A}))^{-1}\|_2 = \|\mathbf{I} + (\mathbf{I} - \mathbf{A}) + (\mathbf{I} - \mathbf{A})^2 + \ldots\|_2 \\
&\leq \|\mathbf{I}\|_2 + \|(\mathbf{I} - \mathbf{A})\|_2 + \|(\mathbf{I} - \mathbf{A})^2\|_2 + \cdots \leq 1 + \sum_{k=1}^{\infty}\left(m\sqrt{\frac{7 \ln n}{2d}}\right)^k \leq 2.
\end{aligned}$$

Finally, we have

$$\|\mathbf{I} - \mathbf{A}^{-1}\|_2 \leq \|\mathbf{A}^{-1}\|_2 \|\mathbf{I} - \mathbf{A}\|_2 \leq m\sqrt{\frac{7\ln n}{2d}}.$$

□

We next provide Theorem 3, which explains why selecting high-scoring (difficult) data samples results in a good performance when the subset size $|\mathbf{S}|$ is large ($m = \omega(d^2 \ln d)$). Assume that we select the subset $\mathbf{X}_\mathbf{S}$ by observing the first element of each data, $(\mathbf{x}_i)_1$. Suppose that we select the data samples whose first elements are $\frac{a_1}{\sqrt{d}}, \frac{a_2}{\sqrt{d}}, \ldots \frac{a_m}{\sqrt{d}}$ where $a_i \in \Theta(1)$, and let $a := \frac{\sum_{i=1}^m a_i^2}{m}$. The elements of the other coordinates are independent normals, i.e., $(\mathbf{x}_i)_k \sim \mathcal{N}(0,1)$ for $k \geq 2$. We will prove that $(\frac{d}{m}\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)^{-1}$ can be approximated by a diagonal matrix of which the first element is equal to $\frac{1}{a}$ and other elements are equal to 1.

Theorem 3 (Sample-sufficient regime)**.** *Let* $\mathbf{B} = \mathrm{diag}(a, 1, 1, \ldots 1) \in \mathbb{R}^{d\times d}$. *If* $m = \omega(d^2 \ln d)$*, then* $\|(\frac{d}{m}\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)^{-1} - \mathbf{B}^{-1}\|_2 \leq c' d^2 \frac{\ln d}{m}$ *for some constant* $c' > 0$ *with high probability as* $d \to \infty$.

Proof. The elements of $\frac{d}{m}\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top$ are expressed as follows, where $k \neq l \in [d]\backslash\{1\}$:

$$\begin{aligned}
\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{11} &= \frac{1}{m}\sum_{i=1}^m a_i^2 = a \\
\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{k1} &= \frac{1}{m}\sum_{i=1}^m a_i \mathcal{N}_i = \mathcal{N}\left(0, \frac{\sum_{i=1}^m a_i^2}{m^2}\right) = \mathcal{N}\left(0, \frac{a}{m}\right) \\
\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{kk} &= \frac{d}{m}\sum_{i=1}^m (\mathbf{x}_i)_k^2 = \frac{1}{m}\mathcal{N}_{ik}^2 = \frac{1}{m}\chi^2(m) \\
\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{kl} &= \frac{d}{m}\sum_{i=1}^m (\mathbf{x}_i)_k(\mathbf{x}_i)_l = \frac{1}{m}\sum_{i=1}^m \mathcal{N}_{ik}\mathcal{N}_{il}.
\end{aligned}$$

By Gaussian tail bound (Lemma A.2), if $\mathbf{x} \sim \mathcal{N}(0, a/m)$, then we have

$$\mathbb{P}(|\mathbf{x}| \geq 2\sqrt{a \ln d/m}) \leq 1/d^2.$$

Note that $\frac{1}{m}\chi^2(m) = \frac{1}{m}\|\mathbf{x}\|_2^2$ for $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I}_m)$. By Lemma A.1, we have a result similar to Eq. 4,

$$\mathbb{P}(|\|\mathbf{x}\|_2 - 1| \geq \sqrt{7\ln d/2m}) \leq 1/d^2.$$

For $\mathcal{N}_{ik}\mathcal{N}_{il}$, by applying the result of Eq. 4, we can also show that

$$\mathbb{P}(\|\mathbf{x}\|_2 \geq \sqrt{7\ln d/2m}) \leq 1/d^3.$$

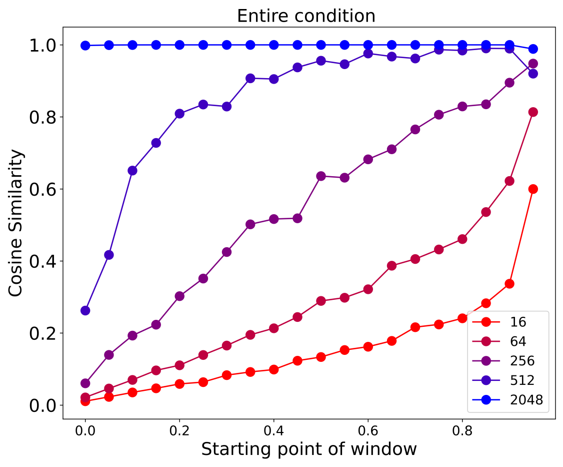
Combining the above three bounds, we obtain that for $\forall k \neq l \in [d]$, with probability $1 - O(\frac{1}{d})$,

$$\left|\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{k1}\right| \leq 2\sqrt{\frac{a\ln d}{m}}, \quad \left|\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{kk} - 1\right| \leq \sqrt{\frac{7\ln d}{2m}}, \text{ and } \quad \left|\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)_{kl}\right| \leq \sqrt{\frac{7\ln d}{2m}}.$$

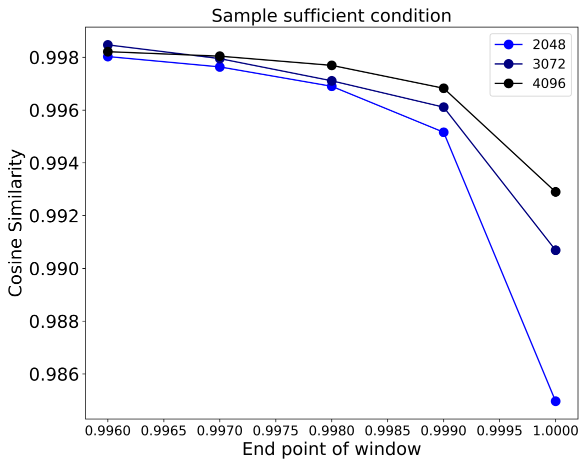
Thus, we obtain $\|\frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top) - \mathbf{B}\|_F \leq cd^2\frac{\ln d}{m}$ for some constant $c > 0$, with probability $1 - O(\frac{1}{d})$. Let $\mathbf{A} = \frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)$, then

$$\begin{aligned}
\|\mathbf{A}^{-1} - \mathbf{B}^{-1}\|_2 &\leq \|\mathbf{A}^{-1}\|_2\|\mathbf{I} - \mathbf{A}\mathbf{B}^{-1}\|_2 \leq \|\mathbf{A}^{-1}\|_2\|\mathbf{B} - \mathbf{A}\|_2\|\mathbf{B}^{-1}\|_2 \\
&\leq \|\mathbf{A}^{-1}\|_2\|\mathbf{B} - \mathbf{A}\|_F\|\mathbf{B}^{-1}\|_2 \leq \|\mathbf{A}^{-1}\|_2 \; cd^2\frac{\ln d}{m} \cdot 1.
\end{aligned}$$

It is remaining to prove that $\|\mathbf{A}^{-1}\|_2$ is bounded. The eigenvalues of $\mathbf{A} = \frac{d}{m}(\mathbf{X}_\mathbf{S}\mathbf{X}_\mathbf{S}^\top)$ are almost equal to the diagonal elements by utilizing Gershgorin circle theorem. Since $m \in \omega(d^2 \ln d)$,



(a) Plots for entire regime



(b) Plots focused on sample-sufficient regime

Figure 7. Results of window sliding experiment at the setting of theoretical analysis. In our setting, the dimension d is 256, the number of full dataset n is $256,000$, and the subset size m are selected among $16, 64, 256, 512, 2048, 3072,$ and 4096. Left figure covers the entire regime ($m = 16, 64, 256, 512,$ and 2048), while the right figure focuses on sample-sufficient regime.

$$r_1 = \sum_{j\neq i} |a_{1j}| < 2d\sqrt{\tfrac{a\ln d}{m}} \ll a \quad \text{and} \quad r_k = \sum_{j\neq i} |a_{kj}| < d\sqrt{\tfrac{7\ln d}{2m}} \ll 1.$$

Therefore, every eigenvalues of $\mathbf{A}$ are close to either a or 1 with probability $1 - O(\frac{1}{d})$. Thus, $\|\mathbf{A}^{-1}\|_2$ is bounded above by $\max(\frac{1}{a}, 1)$ plus some some constant, which shows that $\|\mathbf{A}^{-1}\|_2$ is bounded. Therefore, we have

$$\|\mathbf{A}^{-1} - \mathbf{B}^{-1}\|_2 \leq \|\mathbf{A}^{-1}\|_2 \, cd^2 \frac{\ln d}{m} \leq c'd^2 \frac{\ln d}{m} \quad \text{for some constant } c' > 0.$$

□

A.3. Toy experiment

To validate our theoretical analysis, we conduct a window sliding experiment similar to the one in Sec. 5, at the setting of the theoretical analysis in Sec. A.2, while varying the subset sizes and the starting points of the window subsets at $d = 256$ and $n = 256,000$. The results are shown in Fig. 7. Fig. 7(a) shows the plot for both the sample-deficient and sufficient regimes, including $m = 16, 64, 256, 512, 2048$, while Fig. 7(b) shows focused plots for sample sufficient regime where $m = 2048, 3072, 4096$. The x-axis in Fig. 7(a) is the starting point of the window subset, which identifies the ranking of the hardest sample in the window subset, while that in Fig. 7(b) is the end point of the window subset, which identifies the ranking of the easiest sample in the window subset. The y-axis is the cosine similarity between $\mathbf{w}$ and $\hat{e}_1 = (1, 0, \dots, 0)$, where $\mathbf{w}$ is the solution of the regression problem, and $\hat{e}_1$ is the unit vector with its first coordinate equal to 1, which is the true decision boundary. A higher cosine similarity implies a better solution. Red lines show the results when subset size m is smaller, and the blue or black lines show the result of larger subset sizes.

In the sample-deficient regime where $m \leq d$ (red lines), the cosine similarity increases as the starting point of the window increases, meaning that it is better to use easy samples to learn the linear classifier. On the other hand, in the sample-sufficient regime where $m > d$ (blue and black lines), the cosine similarity is larger for windows having a smaller end point, meaning that it is better to include difficulty samples to learn a better classifier. This result coincides with the theoretical analysis, which claims that the inclusion of easier (harder) data samples results in a better solution for a smaller (larger) subset size, respectively. As we conjectured at Sec. 3.2, the transition of a desirable selection strategy occurs near $m = \Theta(d)$.

B. Discussions on Using Kernel Ridge Regression as a Proxy

In Algorithm 1, we use the kernel ridge regression as a proxy for training neural networks to evaluate the performance of window subsets. In this section, we provide some theoretical rationale behind the use of the kernel ridge regression.

Our use of the kernel ridge regression as a proxy for training neural networks can be partly explained by the recent progress in theoretical understanding of training neural networks using kernel methods. In particular, some recent works (Neal, 1996; Lee et al., 2018; Jacot et al., 2018; Arora et al., 2019) have shown that training and generalization of neural networks can be approximated by two associated kernel matrices: the Conjugate Kernel (CK) and Neural Tangent Kernel (NTK). The Conjugate Kernel is defined by the gram matrix of the derived features produced by the final hidden layer of the network, while NTK is the gram matrix of the Jacobian of in-sample predictions with respect to the network weights. These two kernels also have fundamental relations in terms of their eigenvalue distributions as analyzed in (Fan & Wang, 2020). Our proxy task is motivated by the observation that the kernel regression with these model-related kernels can provide a good approximation to the original model (under some assumptions such as enough width, random initialization, and small enough learning rate, etc.). As an example, the work by Arora et al. (2019) provides the following theorem, which connects the training of a neural network with kernel ridge regression using NTK.

Theorem 4 (Informal version of Arora et al. (2019))**.** *Consider a fully connected neural network with sufficiently large width $d_1 = d_2 = \dots d_L$ where d_l is the number of nodes in lth layer. Given a training dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n \subset \mathbb{R}^d \times \mathbb{R}$ with normalized inputs $\|\mathbf{x}_i\|_2 = 1$, the network is trained by gradient descent with a sufficiently small learning rate to minimize the square-loss $\sum_{i=1}^n (f_{nn}(\mathbf{x}_i) - y_i)^2$, where f_{nn} is the trained network. With a kernel function of the network $K(\cdot,\cdot)$, NTK of training data $\mathbf{H} \in \mathbb{R}^{n\times n}$ is defined by $\mathbf{H}_{ij} = K(\mathbf{x}_i, \mathbf{x}_j)$. And, for a test data $\mathbf{x}_{te}$, the kernel between the test data and the training dataset $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_n] \in \mathbb{R}^{d\times n}$ is defined by $\mathbf{K}(\mathbf{x}_{te}, \mathbf{X}) \in \mathbb{R}^n$ where $\mathbf{K}(\mathbf{x}_{te}, \mathbf{X})_i = K(\mathbf{x}_{te}, \mathbf{x}_i)$. Let $f_{ntk}(\mathbf{x}_{te}) = (\mathbf{K}(\mathbf{x}_{te}, \mathbf{X}))^\top \mathbf{H}^{-1}\mathbf{y}$. Then,*

$$|f_{nn}(\mathbf{x}) - f_{ntk}(\mathbf{x})| \le \epsilon$$

Theorem 4 justifies that the kernel regression with NTK can provide a good approximation to the neural network training under the specified assumptions. However, calculating the NTK for the entire neural network requires high computational cost as it involves computing the Jacobian with respect to the network weights.

To address such a problem, Conjugate Kernel is often considered as a promising replacement of NTK. For example, the work by Zhou et al. (2022) utilizes the kernel ridge regression based on Conjugate Kernel for dataset distillation. We also use the Conjugate Kernel in our kernel ridge regression (Eq. 1), by defining the kernel matrix as $\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S}$ where $\mathbf{X}_\mathbf{S} = [\mathbf{f}_1, \dots, \mathbf{f}_m]$ is composed of features produced by the exact target network of our consideration (ResNet18 for CIFAR-10 and ResNet50 for CIFAR-100/ImageNet). By considering the features from the target network, we can obtain the (approximate) network predictions that are linear in these derived features. In detail, the output of the CK-regression for the test example $\mathbf{x}_{te}$ can be written as $f_{ntk}(\mathbf{x}_{te}) = \mathbf{x}_{te}^\top \mathbf{X}_\mathbf{S}(\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1}\mathbf{y}_\mathbf{S} = \mathbf{x}_{te}^\top \mathbf{w}$ where $\mathbf{w} = \mathbf{X}_\mathbf{S}(\mathbf{X}_\mathbf{S}^\top \mathbf{X}_\mathbf{S})^{-1}\mathbf{y}$ in Eq. 2 for $\lambda = 0$.

Of course, this kernel approximation of the neural network models, which assumes a fixed feature extractor, does not exactly match our situation where the selected subset not only affects the linear classifier but also the feature extractor itself during the training. However, this is still a good proxy that can reflect the network architecture of our interest in a computationally-efficient manner. Also, our analysis in Table 2 shows that this proxy finds the best window subset that aligns well with the result from the actual training of the full model.

C. Implementation Details

C.1. Baseline details

We benchmark our BWS algorithm against eight different state-of-the-art methods, Forgetting score (Toneva et al., 2019), EL2N score (Paul et al., 2021), AdaCore (Pooladzandi et al., 2022), LCMat (Shin et al., 2023), Moderate DS (Xia et al., 2023), CCS (Zheng et al., 2023), SSL Prototype score (Sorscher et al., 2022), and Memorization score (Feldman & Zhang, 2020).

In Forgetting and EL2N scores, the scores are derived by averaging the results of five independent training runs using the full CIFAR-10/100 dataset. Specifically, Forgetting scores are obtained at the 200th epoch (full training), while EL2N scores are captured at the 20th epoch. For our ImageNet experiments, pre-calculated Forgetting, EL2N, SSL prototype, and Memorization scores are sourced from https://github.com/rgeirhos/dataset-pruning-metrics (Sorscher et al., 2022).

In the AdaCore and LCMat methodology, subset selection is conducted only once at the 10th epoch, to ensure a fair comparison to other baselines. Both AdaCore and LCMat implementations are sourced from the LCMat repository. For Moderate DS, models are trained using the full dataset, and the individual data features are extracted from the models. These features are defined as the outputs of the penultimate layer, with dimensions being 512 for ResNet18 and 2048 for ResNet50. The CCS algorithm employs the aforementioned Forgetting score. Within the CCS approach, we consistently set the hyperparameter β to zero across all data selection ratios. We also provide the results of CCS with optimal β obtained by grid search in Appendix §E.

All computational tasks utilized consistent network architectures, as detailed in Section 6: ResNet18 for CIFAR-10 and ResNet-50 for both CIFAR-100 and the ImageNet dataset. Additional experimental specifications related with learning algorithms are reported in Table 5 of Appendix §C.2.

Details of the baselines are summarized below:

- EL2N score: The Error L2-Norm (EL2N) score of data $(\mathbf{x}_i, y_i)$ is defined as $\mathbb{E}[\|f(\mathbf{W}(t), \mathbf{x}_i) - y_i\|_2]$, where $f(\mathbf{W}(t), \mathbf{x})$ is the output of the neural network for the sample $(\mathbf{x}, y)$ at the t-th epoch.

- Forgetting score: The Forgetting score is defined as the number of times during training (until epoch T) that the decision of the sample switches from a correct one to an incorrect one. Forgetting$(\mathbf{x}_i, y_i)$ is defined as

$$\sum_{t=2}^{T} \mathbb{1}\{\arg\max f(\mathbf{W}(t-1), \mathbf{x}_i) = y_i\}(1 - \mathbb{1}\{\arg\max f(\mathbf{W}(t), \mathbf{x}_i) = y_i\}). \tag{8}$$

- AdaCore: Adaptive Second order Coresets (AdaCore) is an algorithm that solves the optimization problem, which finds a subset that imitates the full gradient preconditioned with the Hessian matrix:

$$S^* \in \text{argmin}_{S \subset V} \sum_{i \in V} \min_{j \in S} \|\mathbf{H}_i(w_t)^{-1}\mathbf{g}_i(w_t) - \mathbf{H}_j(w_t)^{-1}\mathbf{g}_j(w_t)\|, \text{ s.t. } |S| \leq r \tag{9}$$

  where $\mathbf{g}_i(w_t) = \nabla l(w_t, (\mathbf{x}_i, y_i))$ and $\mathbf{H}_i(w_t) = \nabla^2 l(w_t, (\mathbf{x}_i, y_i))$ represent the gradient and Hessian of the loss function for the data point $(\mathbf{x}_i, y_i)$ using the model parameter w_t at the t-th epoch of training, respectively. Let V represents the full dataset and S be the coreset of size r. In the AdaCore method, when employing the cross entropy loss with a softmax layer as the final layer, the gradient $\mathbf{g}_i(w_t)$ is approximated by $p_i - y_i$, where p_i is the softmax output for the data point $(\mathbf{x}_i, y_i)$. Moreover, to reduce computational complexity, the Hessian $\mathbf{H}_i(w_t)$ is approximated using only its diagonal.

- LCMat: Loss-Curvature Matching (LCMat) is an algorithm that solves the optimization problem, which finds a subset that matches the loss curvature of full dataset. Due to the intractability of utilizing the loss curvature, an alternative optimization problem is suggested as follows:

$$S^* \in \text{argmin}_{S \subset V} \sum_{i \in V} \min_{j \in S} \|\mathbf{g}_i(w_t) - \mathbf{g}_j(w_t)\| + \frac{1}{2}\rho \sum_{k \in \mathcal{K}} |\lambda_{i,k} - \lambda_{j,k}|, \text{ s.t. } |S| \leq r \tag{10}$$

  where $\mathbf{g}_i(w_t) = \nabla l(w_t, (\mathbf{x}_i, y_i))$ and $\lambda_{i,k} = [\mathbf{H}_i(w_t)]_{kk} = \nabla^2_{kk} l(w, (\mathbf{x}_i, y_i))$ denote the gradient and the k-th diagonal element of the Hessian of the loss function for the data point $(\mathbf{x}_i, y_i)$ with the model parameter w_t at the t-th epoch of training, respectively. Let V represent the full dataset, S the coreset with size r and W the model parameter space. $\mathcal{K} = \arg\max_{|\mathcal{K}|=K} \sum_{j \in \mathcal{K}} \text{Var}_i(\lambda_{i,k})$ is a set of indices for K sub-dimensions on W, where the dimension variance is high. In LCMat, when employing the cross entropy loss with a softmax layer as the final layer, the gradient $\mathbf{g}_i(w)$ is approximated by $p_i - y_i$, where p_i is the softmax output for the data point $(\mathbf{x}_i, y_i)$.

- Moderate DS: For each class of a given dataset, Moderate Coreset calculates the distance between features and the feature mean of the class, which is defined as $d_i = \|\mathbf{f}_i - \frac{\sum_{j \in S} \mathbf{f}_j}{|S|}\|_2$ where S is the set of features whose label is the same as f_i. Then, the data points with distances closest to the distance-median($median(\{d_i\}_{i \in S})$) are selected.

- CCS: Coverage-Centric Coreset Selection (CCS) is an algorithm based on difficulty-based score, which considers overall data coverage upon a distribution as well as important data. CCS prunes $\beta\%$ hardest data first and splits the remained data into k subsets $\{\mathbf{B}_i\}_{i=1}^k$ based on evenly divided score ranges $\{\mathbf{R}_i\}_{i=1}^k$. Then, CCS selects the same number of samples from each score range to make the score distribution of the selected samples uniform.

Table 5. Details for the experiments used in the training of the dataset.

	CIFAR-10	CIFAR-100	ImageNet
Architecture	ResNet18	ResNet50	ResNet50
Batch size	128	128	256
Epochs	200	400	90
Initial Learning Rate	0.05	0.2	0.1
Weight decay	5e-4	5e-4	1e-4
Learning Rate Scheduler	Cosine annealing scheduler		Step scheduler
Optimizer	SGD with momentum 0.9		
Data Augmentation	Random Zero Padded Cropping (4 pixels) Random left-right flipping (probability 0.5) Normalize by dataset's mean, variance		Random Resized Cropping

- SSL Prototype score: The work by Sorscher et al. (2022) conducts k-means clustering of samples in the embedding space of a model pre-trained on the ImageNet dataset. It then defines a self-supervised prototype metric (SSL Prototype score) as the Euclidean distance to its nearest cluster centroid, or prototype. Points located closer to the center have lower SSL scores.

- Memorization score: Memorization score (Feldman & Zhang, 2020) calculates the influence of each training example $(\mathbf{x}_i, y)$ on the classification accuracy of that same example $(\mathbf{x}_i, y)$, and is defined as follow

$$\text{mem}((\mathbf{x}_i, y_i)) = \mathbf{P}(h_T(\mathbf{x}_i) = y_i) - \mathbf{P}(h_{T\setminus\{(\mathbf{x}_i, y_i)\}}(\mathbf{x}_i) = y_i) \tag{11}$$

where $h_S(\cdot)$ is a model trained on the set S and T is the training dataset.

C.2. Experiment details

Data pruning experiment We conduct experiments with three public datasets, CIFAR-10/100 and ImageNet by training ResNet networks (He et al., 2016) of different depths. ResNet18 is used for CIFAR-10 and ResNet50 is used for CIFAR-100 and ImageNet dataset. The implementation is based on the ResNet network in torchvision (Paszke et al., 2019). Since CIFAR-10/100 images are smaller than ImageNet images, we replace the front parts of the ResNet (convolution layer with 7×7 kernel and 2×2 stride, max pooling layer with 3×3 kernel and 2×2 stride) with a single convolution layer with 3×3 kernel and 1×1 stride for the small size images. The details on hyperparameters and optimization methods used in training are summarized in Table 5.

Our experiments report the averaged results from three runs on CIFAR-10/100 and two on ImageNet, with shaded regions representing standard deviations. Networks are trained on datasets curated based on specific selection ratios and methods. Crucially, our data selection ensures equal selection from each class by preserving the portion data in each class.

Cross-architecture robustness We conduct cross-architecture experiments on the CIFAR-10 dataset, training three distinct networks: a simple CNN, EfficientNet-B0, and a Vision Transformer (ViT) pretrained on the ImageNet dataset.

For the simple CNN, we design an architecture comprising three convolutional layers with a 3×3 kernel and 1×1 stride (channels: 64, 128, 256). This is paired with two max-pooling layers with a 2×2 kernel. The convolutional layers are interspersed with these max-pooling layers. Following the convolutional layers, the network is connected to two fully connected layers (channels: 128, 256). Each convolutional layer is equipped with a batch normalization layer followed by a non-linear ReLU activation layer. We set the initial learning rate to 0.05 and weight decay to 1e-4. Other details are the same as CIFAR-10 case in Table 5. For EfficientNet-B0, we closely follow the implementation details of Tan & Le (2019), by setting the learning rate to 1e-4 and a weight decay of the same magnitude. Other implementation specifications are the same with the details in Table 5 of CIFAR-10. For the ViT, we adhere to the implementation specifications as detailed in Dosovitskiy et al. (2021). We obtain a ViT model pretrained on the ImageNet dataset using the timm module in PyTorch, which we subsequently fine-tune on the CIFAR-10 dataset for 10 epochs. For fine-tuning a model pre-trained on ImageNet to adapt to the CIFAR-10 dataset, we resize the data to fit the 224x224 pixel dimensions. We set the initial learning rate to

1e-4 and weight decay to 1e-4. We do not use a learning rate scheduler. Other details are the same as CIFAR-10 case in Table 5.

Within our algorithm, BWS, we utilize a Forgetting score sourced from the ResNet18 architecture. Furthermore, we establish a feature extractor using either simple CNN, EfficientNet-B0, or ViT architecture, repectively. For the CNN and EfficientNet-B0 architecture, we execute training for 20 epochs, while for the ViT setup, we fine-tune for 3 epochs. We report the averaged results from three independent runs on the three networks, with the shaded regions indicating the standard deviations. Similar to previous experiments, data samples are selected to ensure a balanced portion of each class, preserving the original class ratios within the CIFAR-10 dataset.

Robustness to label noise We generate a noise version of the CIFAR-10 dataset with symmetric label noise at levels of 20% and 40%, respectively. To evaluate this noisy dataset, we compute the EL2N score using a ResNet18 model, averaging the outcomes over five independent runs. The EL2N score is selected due to its lower computational cost compared to the Forgetting score, especially when it is required to re-calculate the new EL2N score for the entire noisy dataset. In our analysis, we apply Algorithm 1 to the CIFAR-10 dataset, where the samples are ordered by their EL2N scores. To calculate the classification accuracy of $\mathbf{w_S}$, we specifically use the lower-scoring 50% of the samples, represented as $\frac{1}{n}\sum_{i=\frac{n}{2}}^{n}\mathbb{1}(\arg\max_c(\mathbf{w}_\mathbf{S}^\top\mathbf{x}_i)_c = y_i)$, instead of the typical approach $\frac{1}{n}\sum_{i=1}^{n}\mathbb{1}(\arg\max_c(\mathbf{w}_\mathbf{S}^\top\mathbf{x}_i)_c = y_i)$. This adjustment was made to exclude noisy samples from the quality evaluation of window subsets and thus prevent overfitting to noise in the data.

Ablation on different window selection methods The formal definitions of Gradient difference and Gradient similarity are as follows:

- Gradient difference: minimizing the difference between the gradients of the full training dataset (V) and window subset (S).

$$\text{Gradient Difference}(V,S) = \left\| \frac{\sum_{i\in V}\nabla f_\mathbf{w}(\mathbf{x}_i)}{|V|} - \frac{\sum_{i\in S}\nabla f_\mathbf{w}(\mathbf{x}_i)}{|S|} \right\|_2 \tag{12}$$

- Gradient similarity: maximizing the cosine similarity between the gradients of the full training dataset (V) and window subset (S).

$$\text{Gradient Similarity}(V,S) = \frac{\sum_{i\in V}\nabla f_\mathbf{w}(\mathbf{x}_i)}{\left\|\sum_{i\in V}\nabla f_\mathbf{w}(\mathbf{x}_i)\right\|_2} \cdot \frac{\sum_{i\in S}\nabla f_\mathbf{w}(\mathbf{x}_i)}{\left\|\sum_{i\in S}\nabla f_\mathbf{w}(\mathbf{x}_i)\right\|_2} \tag{13}$$

C.3. Computational cost

Table 6. Time cost (in seconds) for subset selection of each algorithm across selection ratios.

Selection ratio		1%	5%	10%	20%	30%	40%	50%	75%	90%
CIFAR-10	BWS (Ours)	4.3	4.8	7.2	8.5	9.7	10.4	10.4	9.0	6.3
	LCMat	520	1197	2260	4213	5800	7173	8450	10320	10631
	AdaCore	224	699	1256	2273	3186	3977	4649	5637	5839
CIFAR-100	BWS (Ours)	14.6	55.6	57.3	62.2	64.7	64.0	61.7	45.7	30.2
	LCMat	1465	1468	1471	1478	1483	1489	1493	1501	1504
	AdaCore	1295	1300	1304	1309	1315	1320	1324	1331	1334
ImageNet	BWS (Ours)	1423	2910	3941	6141	7590	8616	9029	10118	6499
	LCMat	238451	239027	239694	240934	242003	242864	243594	244854	245245
	AdaCore	213733	214309	214963	216181	217067	217866	218521	219593	219924
GPU		Nvidia A100 40GB								

We compare the computational cost of our BWS algorithm, as detailed in Algorithm 1, with other optimization-based coreset selection baselines, namely LCMat and AdaCore. We assume that the sample scores, used for sorting, is readily available and that the feature extractor is also pre-provided for both ours and optimization-based methods. We report and compare the time taken to select the subset for each algorithm.

In Table 6, we detail the time required to select subsets from various datasets using the different methods. Clearly, our method outperforms optimization-based techniques in terms of time cost for subset selection. As we have previously described in Sec. 5, our strategy, which selects the best window subset from a continuous interval of samples sorted by their scores, greatly reduces the search space compared to the general optimization techniques, leading to improved efficiency.

D. Detailed Review of Related Works

In this section, we provide additional related works on data subset selection regarding various perspectives.

When there is no validation set, some score-based selection methods, such as EL2N (Paul et al., 2021) and Forgetting (Toneva et al., 2019), suffer from performance degradation when the dataset includes label-noise samples, since these methods often assign high scores to label-noisy samples, as label-noise samples are inherently hard to learn. Some recent methods adopt more cautious measures to distinguish hard-to-learn but clean-label samples, known to be valuable to enhance the generalization ability of neural networks, from label-noise samples. For instance, Cartography (Swayamdipta et al., 2020) utilizes two measures, confidence mean and confidence variance of data sample, to distinguish hard-to-learn samples from mere label-noise samples. Second-Split Forgetting (Maini et al., 2022) achieves this goal by observing the learning time and forgetting time of each sample while training a model. AUM (Pleiss et al., 2020) observes the logit value of a given label and the next largest logit value, and uses the gap to separate noisy data samples and ambiguous data samples.

Another important issue that has been recently explored in data subset selection is the computational overhead in quantifying the data value. There are several recent attempts to valuate data without training of a neural network. For example, CG-score (Ki et al., 2023) evaluates data instances without model training by calculating the analytical gap in generalization errors when an instance is held out. LAVA (Just et al., 2023) evaluates the value of each data instance without a model training by using a proxy function, the class-wise Wasserstein distance between training and validation set, for the validation performance. DAVINZ (Wu et al., 2022) utilizes the Neural Tangent Kernel (NTK) of a network at initialization for calculating the contribution of each data instance to domain-aware generalization error bound.

In optimization-based selection, many works have utilized the effect of data on the model training. MaxMargin (Har-Peled et al., 2007), IWeS (Citovsky et al., 2023), and Selection-Via-Proxy (Coleman et al., 2020) use the confidence of a model to identify uncertain data during optimization. Maximum Margin Coresets (Har-Peled et al., 2007) selects data with the smallest margin in an SVM setting, IWeS (Citovsky et al., 2023) selects examples using importance sampling with a sampling probability based on the confidences of two models, and Selection-Via-Proxy (Coleman et al., 2020) applies confidence-based methods to a small proxy model to perform data selection. Das et al. (2021) solve a convex linear programming problem to find high-value data that contributes much to the loss and optimization, and Glister (Killamsetty et al., 2021b) finds data that contributes significantly to the loss during the training of a neural network. GradMatch (Killamsetty et al., 2021a) finds a subset whose gradient matches better with the gradient of the full dataset.

Additionally, in active learning, where data samples are selectively labeled for semi-supervised learning, Neural-Preconditioning (Kong et al., 2022) obtains the label of data that dominates the eigenspace of the NTK, and Culotta & McCallum (2005) obtain the label of the least confident data to reduce labeling costs. The works by Har-Peled et al. (2007); Citovsky et al. (2023); Coleman et al. (2020) utilize the confidence of a model to identify uncertain data during the optimization, and the works by Das et al. (2021); Killamsetty et al. (2021b) find the data that contributes much to the loss.

E. Comparison with CCS

CCS (Zheng et al., 2023) prunes the hardest β% samples, divides the remaining data into non-overlapping k ranges based on the difficulty scores, and uniformly assigns budgets to each range. Samples are then chosen from each range within the budget. If a range has fewer data than the assigned budget, the remaining budget is iteratively reassigned to other ranges. While methodologies based on difficulty scores suffer from a drastic performance drop at low subset ratios, CCS achieves high performance across a broad range by selecting diverse data with appropriate β and k. However, CCS does not propose an efficient method to find the desired β and reports the result with the optimal β obtained by grid search, which may require high computational cost. Since choosing the best performing model among the models trained on each subset chosen by different β is not fair for comparison to other baselines, including BWS, we reported the result of CCS obtained by setting the hyperparameter β as 0 in the main experimental results. In this section, we additionally report the results of CCS with the optimal β found by grid search and compare the performance with those of BWS and oracle window in Table 9. In Table 8, we also report the optimal β for CCS across different selection ratios. For the subset ratios of 10%, 20%, 30%, and 50%, we utilize the β values reported in the original paper (marked with †), and for other subset ratio, we conduct a grid search to find the best β by exploring it with a step size of 10%.

Several key observations emerge from the results. First, CCS after hyperparameter tuning, achieves performance comparable to the Oracle Window at lower selection ratios (1-10%). This is a natural consequence, given both methods' ability to discard the top hardest samples in favor of easier ones at low ratios. However, at higher ratios (20%-90%), CCS's efficacy decreases relative to both Oracle Window and BWS, even after tuning β. This decline can be attributed to CCS's strategy of selecting samples across a uniform score distribution after pruning the hardest ones. Even CCS adjusts β to 0 or lower values (e.g., 10 or 20) for ratios beyond 30%, the sample selection with uniform score distribution makes CCS incorporate not only hard samples but also easy samples, which are less effective in high subset ratios. In comparison, Optimal Window or BWS, which select samples from a contiguous difficulty range, focus on selecting harder samples as the subset ratio increases, resulting in better performance. Moreover, we analyze the computational costs associated with CCS and BWS as summarized in Table 7. Since CCS requires the repeated training of deep neural networks in the process of tuning the hyperparameter β, it requires significant computational overhead compared to BWS. On the other hand, BWS circumvents the need for hyperparameter tuning, by solving a simple proxy task to identify the best window subset, which considerably shortens the time requirement.

Table 7. Time cost (in seconds) to compute CCS and BWS.

Selection ratio		5%	30%	75%
CIFAR-10	CCS	812	3897	3654
	BWS	13	59	130
CIFAR-100	CCS	3909	18767	17594
	BWS	75	182	339

Table 8. Optimal β (%) at different selection ratios in various dataset.

Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%
CIFAR-10	80	50	$30^\dagger$	$10^\dagger$	$10^\dagger$	10	$0^\dagger$	0	0
CIFAR-100	99	80	$50^\dagger$	$40^\dagger$	$20^\dagger$	20	$20^\dagger$	20	0
ImageNet	80	30	$30^\dagger$	$20^\dagger$	$20^\dagger$	10	$10^\dagger$	10	0

Table 9. Test accuracy of CCS with optimal β at different selection ratios.

Selection ratio		1%	5%	10%	20%	30%	40%	50%	75%	90%
CIFAR-10	CCS	46.58	72.12	81.99	88.89	91.74	93.10	94.09	94.95	95.29
	Oracle	47.17	72.89	82.67	89.06	91.80	93.59	94.54	95.23	95.37
	BWS	46.10	70.70	82.29	88.74	91.80	93.59	94.54	95.23	95.37
CIFAR-100	CCS	11.01	31.11	45.52	56.09	64.26	68.51	70.80	75.83	78.13
	Oracle	10.63	30.39	45.16	58.91	67.51	72.70	75.00	78.42	79.00
	BWS	8.43	29.25	44.11	58.30	67.20	72.17	73.83	77.78	79.00
ImageNet	CCS	8.13	31.40	45.10	57.12	62.65	67.54	69.58	73.10	74.59
	Oracle	7.97	33.58	48.84	62.83	68.22	71.33	72.74	74.73	75.25
	BWS	7.02	33.31	46.72	62.32	67.16	70.47	72.68	74.73	75.25

F. Additional Experiments

F.1. Sliding window experiment for ImageNet dataset

We investigate the efficacy of the window selection approach by varying the starting points and demonstrate the existence of an optimal window subset. In this process, we arrange the ImageNet samples in descending order based on their Forgetting

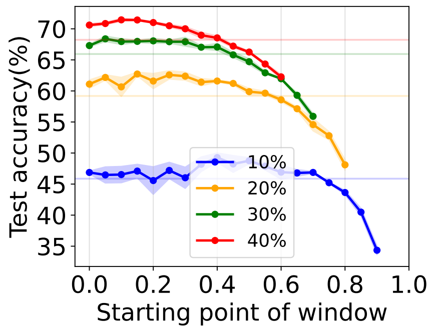


Figure 8. Sliding window experiments in ImageNet dataset to measure the test accuracy of the models trained by window subsets while changing the starting point of the windows. Samples are sorted in descending order by their difficulty scores. The horizontal lines are results from random selection. For each subset ratio, there exists the best window, and its starting point shifts toward left as the subset ratio increases.

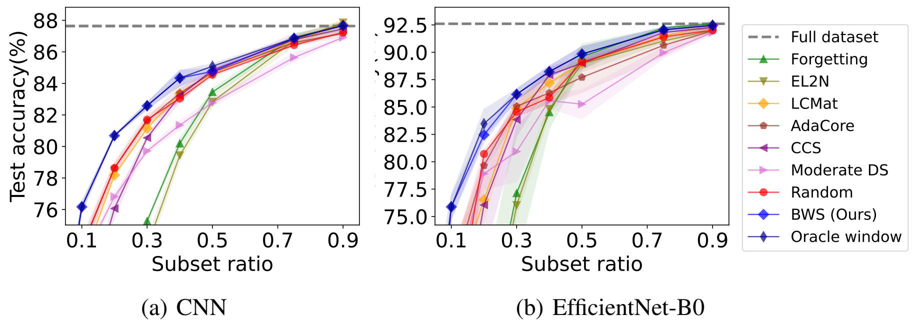


Figure 9. Cross-architecture experiments with CNN (left), and EfficientNet-B0 (right), where samples scores are calculated using ResNet18 model. Full results are reported in Table 16–17.

scores (Toneva et al., 2019), and then select windows of varying sizes, from 10% to 40%. The starting point for these windows is adjusted from 0 to $(100 - w)\%$, incrementing in steps of 5%. Subsequently, we train a ResNet50 model using these window subsets and present the resulting test accuracies in Fig. 8. Consistent with the observations in Fig. 3, within the ImageNet dataset, we note that for each subset ratio, there is an optimal starting point. Notably, this optimal point shifts progressively towards lower values, which correspond to more difficult samples, as the size of the window subset increases.

F.2. Cross architecture robustness

To test the robustness of our method across changes in neural network architectures, we conduct data pruning experiments on CIFAR-10 while using different architectures during sample scoring and training. The window subsets are constructed using samples ordered by their Forgetting scores, calculated on ResNet18 architecture. Then, the best window selection (Alg. 1) and the model training are conducted using a simpler CNN architecture or EfficientNet-B0 architecture. The results on the CNN architecture are presented in Fig. 9(a), and those on the EfficientNet-B0 are shown in Fig. 9(b). In all cases, our method (BWS) consistently achieves competitive performances across all selection ratios, demonstrating its robustness to changes in neural network architectures during data subset selection.

F.3. Robustness to label noise

We test the robustness of BWS in the presence of label noise in the training dataset. We corrupt randomly chosen 20% and 40% samples of CIFAR-10 by random label noise. It has been previously reported that the difficulty score-based selection methods are susceptible to label noise since such methods tend to assign high scores to label-noise samples (Toneva et al., 2019; Paul et al., 2021). Thus, these methods often ends up prioritizing the label-noise samples in the selection process, leading to suboptimal results. On the other hand, our algorithm offers flexibility in choosing window subsets with varying

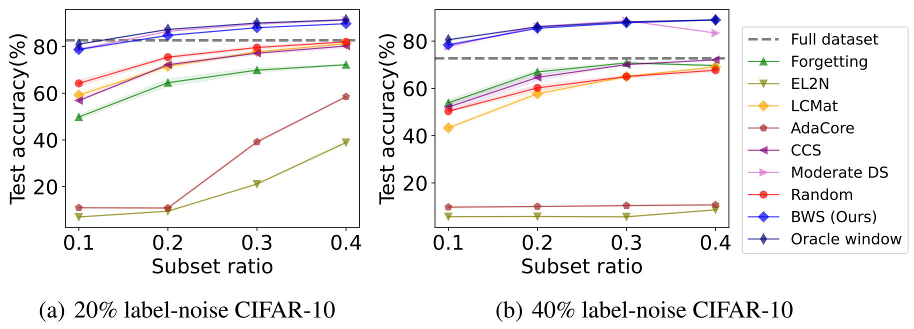


Figure 10. Data pruning experiments with CIFAR-10, including (a) 20% label-noise, (b) 40% label-noise. Our method (BWS) attains the accuracy of the full training dataset despite the presence of label noise. Full results are reported in Table 19-20.

levels of difficulty by changing the starting point, and adopts an approach to select the best window by solving a proxy task using the kernel ridge regression. To further enhance the robustness of our method, we can modify Alg. 1 to evaluate the solution of kernel ridge regression using only the low-scoring 50% samples from the training dataset, which will rarely include label-noise samples, instead of the full dataset. We use EL2N (Paul et al., 2021) as the difficulty score to align the samples in our algorithm. In Fig. 10, we compare the performance of this modified version of BWS with other baselines. While difficulty score-based selection and optimization-based selection methods suffer from performance degradation due to label noise, our method, along with another label noise-robust method, Moderate DS, achieves performance even higher than what is achievable with the full training dataset, which includes the 20% or 40% label noise, respectively.

G. Ablation studies

G.1. Ablation study on window type: two half-width sliding windows

BWS sorts samples in a dataset by their difficulty scores and then selects the optimal window subset from one continuous single-interval regime. Thus, the window selection chooses the samples of similar difficulty level. To further examine possible benefits from non-contiguous subset selection, we conduct an additional experiment on the CIFAR-10 dataset by finding the optimal two half-width windows while varying their starting points. In detail, we sort the samples from CIFAR-10 in descending order based on Forgetting score (Toneva et al., 2019) and for a subset of size $w\%$, we search over all combinations of two half-width windows, denoted by $[x_1, x_1 + w/2] \cup [x_2, x_2 + w/2]$ while varying their starting points (x_1, x_2) in $x_1 \in [0, 100 - w]$ and $x_2 \in [x_1 + w/2, 100 - w/2]$ with a step size of 5%. We train ResNet18 on each subset and evaluate the corresponding test accuracies. The full results are presented in Fig. 11, and in Table 10 we report the top five results (the compositions of half-width windows and their test accuracies) for subset ratios ranging from 10 to 40%. We highlight the cases where the two half-width windows are contiguous to each other with bold letters.

Table 10. Top-five test accuracies and their corresponding half-width window compositions on CIFAR-10 dataset. We highlight the cases where the two half-width windows are contiguous to each other with bold letters.

Ratio	Ranking	1st	2nd	3rd	4th	5th
10%	Half-width windows	40-45%, 50-55%	40-45%, 55-60%	45-50%, 55-60%	**50-55%, 55-60%**	**45-50%, 50-55%**
	Test Acc	83.04	82.87	82.71	**82.67**	**82.46**
20%	Half-width windows	20-30%, 35-45%	**25-35%, 35-45%**	20-30%, 40-50%	20-30%, 50-60%	25-35%, 45-55%
	Test Acc	89.16	**89.06**	88.98	88.84	88.77
30%	Half-width windows	10-25%, 30-45%	10-25%, 35-50%	5-20%, 30-45%	15-30%, 35-50%	**15-30%, 30-45%**
	Test Acc	92.02	91.98	91.90	91.84	91.80
40%	Half-width windows	5-25%, 30-50%	**5-25%, 25-45%**	10-30%, 35-55%	5-25%, 35-55%	5-25%, 40-60%
	Test Acc	93.67	**93.59**	93.54	93.46	93.40

We can observe that for every considered subset ratio, the top-five best performing cases include contiguous windows or windows near to each other with the gap of only 5%, even though we allowed flexibility in choosing the two half-width

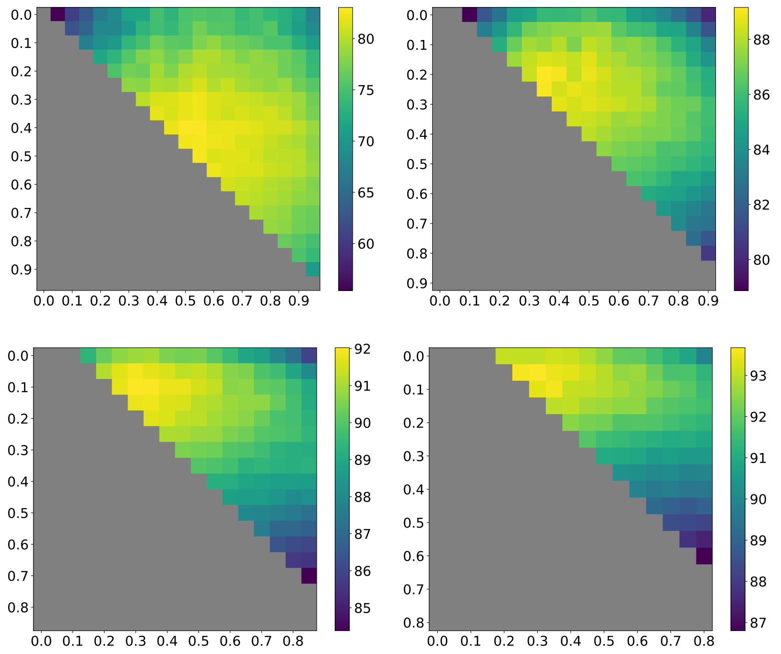


Figure 11. Test accuracy of the models trained with two half-width windows of subset ratios 10% (top left), 20% (top right), 30% (bottom left), and 40 %(bottom right) with varying starting points. The numbers in axes indicate the starting points of each interval, and the color indicates the test accuracy for each composition of half-width windows. We note that a contiguous window, a point near the diagonal, attains performance levels comparable to the best performance.

windows far away from each other. This result further supports our use of window selection, which only considers subsets from a continuous interval of samples based on difficulty scores, in choosing near-optimal subset in an efficient manner across a broad range of selection ratios.

G.2. Ablation study on window type: wider sliding windows

We also conduct an additional experiment on the CIFAR-10 dataset to explore non-contiguous sample selection by considering random selection from wider windows. By arranging the samples in descending order according to difficulty scores and selecting a starting point, denoted as $s\%$, for a given subset ratio of $w\%$, we randomly choose samples within the range $[s, s + c \cdot w]\%$, where c is a constant greater than 1. In particular, we sort the CIFAR-10 samples in descending order based on their Forgetting scores (Toneva et al., 2019), and then select windows of various sizes, ranging from 10% to 40%, by adjusting the starting point from 0 to $(100 - cw)\%$ in 5% increments. The window width is cw for a given ratio $w\%$ and a constant c equal to either 1.5 or 2. Subsequently, we randomly select $w\%$ of the data from the window, train the ResNet18 network with this subset, and plot the resulting test accuracies in Fig. 12.

We observe that training with a wider window, regardless of the constant c, results in test accuracy curves similar to those shown in Fig. 3(a). However, the sliding window experiment for wider windows in Fig. 12 shows that the best contiguous window (horizontal lines) achieves better performance than wider windows, especially in high ratios. Thus, this result supports the use of a contiguous window subset in sample selection across a broad range of selection ratios.

G.3. Ablation study on difficulty scores

In the implementation of our BWS algorithm, we employ the Forgetting score (Toneva et al., 2019) as a difficulty score. To test the algorithm's adaptability to alternative difficulty scores, we examine its performance when configured with the EL2N score (Paul et al., 2021) and C-score (Jiang et al., 2021). Table 11 presents a comparison of the results obtained by our BWS algorithm when utilizing the EL2N score and C-score, against those achieved with the Forgetting score. Regardless of the difficulty score used, all the results demonstrate competitive performances, closely approaching those of the oracle window, across a wide range of selection ratios. We anticipate that the observed phenomenon arises due to a strong correlation

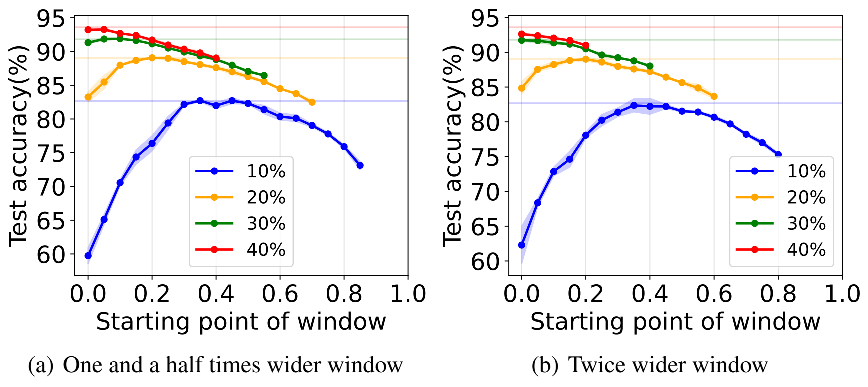


Figure 12. Test accuracy using subsets randomly sampled from windows of (a) one and a half times ($\times 1.5$) and (b) twice ($\times 2$) larger than the subset ratio, while varying the starting points of these windows. The horizontal lines represent the results from the oracle window, which is the maximum test accuracy obtained in sliding window experiments, for each subset ratio. Our observations indicate that at lower subset ratios, there are subsets whose performance is comparable to that of the oracle window, but, at higher subset ratios, the performance of all subsets consistently falls short of the oracle window.

Table 11. Test accuracy of the BWS algorithm at different data selection ratios, depending on the difficulty score. Due to the high correlation between the difficulty scores, there is a similar sorting order across the scores. Thus, similar window positions are selected by BWS, regardless of the specific difficulty score in use. This similarity in subset selection leads to consistently strong performance regardless of the chosen difficulty score.

Selection methods	Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%
BWS with C-score	Test accuracy	46.25	71.34	82.02	89.12	91.85	93.62	94.62	95.18	95.28
	Window index	85%	55%	45%	25%	15%	5%	0%	0%	0%
BWS with EL2N	Test accuracy	45.02	71.87	81.79	88.87	91.59	93.39	94.44	95.06	95.32
	Window index	80%	60%	45%	25%	10%	5%	0%	0%	0%
BWS with forgetting (Ours)	Test accuracy	46.10	70.70	82.29	88.74	91.80	93.59	94.54	95.23	95.37
	Window index	90%	70%	55%	30%	15%	5%	0%	0%	0%
Oracle window	Test accuracy	47.17	72.89	82.67	89.06	91.80	93.59	94.54	95.23	95.37
	Window index	85%	55%	50%	25%	15%	5%	0%	0%	0%

between the difficulty scores. The rank correlation between the EL2N score (C-score) and the forgetting score used for comparison is notably high as 0.8836 (0.8500). This suggests that samples sorted by the different difficulty scores would likely follow a similar order of forgetting score. As a result, the best windows selected by BWS for the two different score cases exhibit similarity, as shown by Table 11. This consistency shows that the effectiveness of BWS is not limited by the choice of the difficulty score, highlighting its robustness to the sample scores used in sorting.

H. Full Results

Table 12 compares the starting points of the window subsets selected by BWS and those of the oracle window subsets. BWS finds the nearly optimal subsets at broad subset ratios for CIFAR-10/100 and ImageNet dataset. In Table 13-20, the oracle window achieves the highest performance among the considered methodologies for almost every selection ratio and dataset, since it finds the best window by directly measuring and comparing the test accuracy of models trained by each window using the test dataset. Since the oracle window cannot be implemented in practice due to significant computational overhead, it is fair to compare the performances among the methods except the oracle window. Thus, we highlight the highest and the second-highest values among the rest of the methodologies except the oracle window in the tables.

Table 12. The starting points (%) of the window subsets obtained by BWS, and those of the oracle window subsets. BWS finds the nearly optimal subsets at broad subset ratios for all datasets.

Selection ratio		1%	5%	10%	20%	30%	40%	50%	75%	90%
CIFAR-10	BWS(Ours)	90	70	55	30	15	5	0	0	0
	Oracle window	85	55	50	25	15	5	0	0	0
CIFAR-100	BWS(Ours)	85	95	75	60	25	10	0	0	0
	Oracle window	95	85	80	55	40	20	5	5	0
ImageNet	BWS(Ours)	90	75	70	5	0	0	0	0	0
	Oracle window	85	65	40	15	5	10	10	0	0

Table 13. Test accuracy of CIFAR-10 dataset on ResNet18. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%	100%
Forgetting	30.08±1.79	42.39±1.31	54.31±0.23	79.19±0.26	89.13±0.17	93.41±0.05	94.49±0.02	**95.31±0.08**	95.14±0.04	
EL2N	15.27±0.36	27.01±0.76	41.27±0.62	71.67±0.82	87.17±0.48	93.24±0.06	94.43±0.13	95.13±0.05	95.26±0.11	
LCMat	41.53±0.61	66.86±1.00	77.48±1.62	87.34±0.22	90.72±0.06	92.45±0.05	93.38±0.07	94.90±0.06	95.19±0.01	
AdaCore	39.87±0.75	66.40±1.10	77.84±0.49	86.88±0.05	89.90±0.08	91.48±0.24	92.73±0.17	94.47±0.18	95.04±0.23	
CCS	31.86±0.72	58.89±1.43	72.61±3.59	86.64±0.35	90.94±0.55	93.00±0.05	94.09±0.17	94.95±0.21	95.29±0.09	95.40±0.08
Moderate DS	40.67±0.50	67.53±0.75	76.62±1.29	84.86±0.09	88.46±0.07	90.63±0.01	91.52±0.08	93.69±0.21	94.68±0.07	
Random	39.10±0.14	67.14±0.29	78.43±0.72	86.87±0.31	89.91±0.31	91.66±0.06	92.83±0.04	94.40±0.05	95.08±0.19	
BWS (Ours)	**46.10±2.68**	**70.70±0.53**	**82.29±0.35**	**88.74±0.18**	**91.80±0.03**	**93.59±0.17**	**94.54±0.06**	95.23±0.08	**95.37±0.07**	
Oracle window	47.17±0.25	72.89±1.05	82.67±0.43	89.06±0.34	91.80±0.03	93.59±0.17	94.54±0.06	95.23±0.08	95.37±0.07	

Table 14. Test accuracy of CIFAR-100 dataset on ResNet50. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%	100%
Forgetting	7.01±0.50	20.69±1.13	34.22±1.27	50.95±0.78	61.54±1.06	68.92±0.87	**73.84±0.95**	**78.55±0.44**	**79.69±0.19**	
EL2N	3.40±0.12	8.15±0.17	14.06±0.48	28.14±1.21	48.13±1.77	52.25±5.85	71.72±0.17	77.33±0.70	78.96±0.10	
LCMat	**8.43±0.44**	28.51±0.65	42.81±0.31	55.77±1.45	64.39±1.02	67.22±0.96	73.11±0.81	77.51±0.37	78.47±0.65	
AdaCore	5.56±0.14	22.76±1.20	39.56±2.53	56.81±1.60	65.30±0.64	70.51±0.64	71.18±1.00	76.62±0.47	78.37±0.32	
CCS	7.49±0.66	24.34±0.35	40.81±2.11	56.81±1.81	63.35±0.40	67.70±0.64	71.04±0.52	74.94±0.73	78.13±0.31	78.81±0.13
Moderate DS	6.05±0.29	24.53±1.28	42.23±3.03	54.72±1.76	64.71±1.27	68.71±2.45	72.61±0.31	75.80±0.48	78.48±0.13	
Random	5.89±0.52	23.76±1.12	42.03±1.56	55.03±1.17	65.98±0.50	69.23±1.04	72.37±0.13	76.53±0.52	78.29±0.22	
BWS (Ours)	8.43±0.49	**29.25±1.15**	**44.11±3.13**	**58.30±0.65**	**67.20±1.67**	**72.17±0.42**	73.83±0.72	77.78±0.55	79.00±0.29	
Oracle window	10.63±0.87	30.39±2.87	45.16±1.28	58.91±0.52	67.51±1.22	72.70±0.50	75.00±0.23	78.42±0.14	79.00±0.29	

Table 15. Test accuracy of ImageNet dataset on ResNet50. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%	100%
Forgetting	4.78±0.10	28.18±0.46	45.84±0.67	60.75±0.60	**67.48±0.11**	70.26±0.48	**72.73±0.09**	74.63±0.13	**75.53±0.06**	
EL2N	2.10±0.08	9.80±0.03	20.42±0.47	41.14±0.04	54.42±0.39	63.19±0.29	68.19±0.13	73.91±0.36	74.79±0.27	
Memorization	0.52±0.04	9.70±0.21	23.80±0.31	44.58±0.09	59.66±0.06	65.92±0.04	70.22±0.02	74.56±0.24	74.94±0.16	
SSL Prototype	1.33±0.21	20.07±1.39	37.98±0.08	55.25±1.02	61.97±0.25	66.58±0.28	68.85±0.19	73.43±0.29	74.63±0.28	
LCMat	6.01±0.31	32.26±0.84	46.08±0.64	59.02±0.36	65.28±0.21	68.50±0.56	70.30±0.46	74.13±0.12	74.81±0.02	
AdaCore	6.01±0.44	31.52±0.58	46.98±0.80	59.26±1.58	65.18±0.05	68.28±0.05	70.72±0.04	73.53±0.13	74.69±0.00	75.85±0.07
CCS	5.04±0.40	31.83±0.62	46.64±1.08	58.77±0.80	64.85±0.12	67.82±0.24	69.89±0.24	73.57±0.12	74.59±0.03	
Moderate DS	5.97±0.60	32.47±0.21	**47.83±0.11**	58.86±0.14	64.71±0.01	67.47±0.03	69.73±0.08	73.16±0.25	74.67±0.07	
Random	6.14±0.01	33.17±0.11	45.87±0.07	59.19±0.04	65.94±0.38	68.23±0.00	70.14±0.31	73.74±0.14	74.83±0.08	
BWS (Ours)	**7.61±0.84**	**33.96±1.08**	46.64±0.20	**62.08±0.51**	67.28±0.20	**70.53±0.16**	72.63±0.14	**74.67±0.10**	75.28±0.28	
Oracle window	7.89±0.27	33.98±0.57	49.21±0.76	62.62±0.30	68.27±0.56	71.35±0.24	72.91±0.38	74.67±0.10	75.28±0.28	

Table 16. Test accuracy of CIFAR-10 dataset by training a simple CNN architecture. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%	100%
Forgetting	34.01±0.45	46.28±0.67	55.04±0.55	67.98±0.29	75.27±0.08	80.19±0.31	83.45±0.23	**86.92±0.17**	87.66±0.29	
EL2N	16.60±0.86	30.58±0.27	42.90±0.18	62.90±0.15	73.67±0.50	79.43±0.36	82.83±0.28	86.72±0.35	**87.86±0.09**	
LCMat	46.42±0.23	65.74±0.65	72.54±0.42	78.19±0.07	81.15±0.11	83.36±0.05	84.65±0.01	86.70±0.29	87.73±0.26	
AdaCore	46.72±0.21	66.69±0.43	73.52±0.49	78.64±0.27	81.62±0.16	83.38±0.49	84.71±0.05	86.59±0.21	87.17±0.22	
CCS	39.50±0.96	59.86±0.21	68.89±0.44	76.07±0.57	80.55±0.14	83.21±0.44	**84.85±0.08**	86.78±0.37	87.46±0.21	87.64±0.14
Moderate DS	48.68±0.46	66.61±0.29	72.64±0.25	76.82±0.28	79.72±0.17	81.35±0.28	82.78±0.24	85.64±0.27	86.91±0.10	
Random	46.70±0.91	66.27±0.48	73.65±0.53	78.63±0.37	81.70±0.36	83.04±0.16	84.55±0.10	86.43±0.05	87.23±0.09	
BWS (Ours)	**52.48±0.42**	**69.71±0.37**	**76.17±0.30**	**80.69±0.06**	**82.58±0.20**	**84.34±0.49**	84.76±0.22	86.86±0.05	87.68±0.06	
Oracle window	53.08±0.29	69.71±0.37	76.17±0.30	80.69±0.06	82.59±0.20	84.34±0.49	85.09±0.16	86.86±0.05	87.68±0.06	

Table 17. Test accuracy of CIFAR-10 dataset by training EfficientNet-B0 architecture. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	1%	5%	10%	20%	30%	40%	50%	75%	90%	100%
Forgetting	26.22±0.74	31.84±2.06	39.56±3.81	62.36±6.14	77.14±6.16	84.53±1.51	89.60±1.15	**92.23±0.06**	**92.60±0.22**	
EL2N	14.67±0.28	24.97±0.84	32.32±2.63	60.11±2.11	76.03±0.88	84.82±0.56	88.95±0.74	91.04±0.15	92.01±0.40	
LCMat	29.12±2.16	54.45±4.23	67.35±4.97	76.52±1.82	84.88±0.53	87.20±1.08	89.02±0.16	91.47±0.20	92.46±0.31	
AdaCore	31.50±2.29	52.96±5.90	69.13±2.58	79.65±0.45	85.06±0.93	86.29±0.96	87.71±1.37	90.64±0.99	91.95±0.10	
CCS	26.38±1.58	45.78±4.05	59.21±6.03	76.05±4.40	83.85±1.55	87.93±0.72	88.99±0.52	91.80±0.20	92.27±0.38	92.60±0.12
Moderate DS	35.10±0.78	55.24±2.45	68.01±3.00	78.91±1.42	80.93±2.77	85.59±0.66	85.26±1.32	89.96±0.48	91.81±0.21	
Random	34.34±3.70	48.49±8.17	62.80±8.18	80.71±0.37	84.56±0.46	85.86±0.47	89.10±0.16	91.40±0.18	92.02±0.24	
BWS (Ours)	**35.86±3.15**	**58.06±4.62**	**75.88±1.07**	**82.46±0.67**	**86.16±0.49**	**88.23±0.50**	**89.84±0.69**	92.06±0.21	92.45±0.29	
Oracle window	42.65±0.66	64.18±4.56	75.88±1.07	83.47±1.32	86.16±0.49	88.23±0.50	89.84±0.69	92.06±0.21	92.45±0.29	

Table 18. Test accuracy of CIFAR-10 dataset by fine-tuning ViT pretrained on ImageNet.We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	1%	5%	10%	20%	100%
Forgetting	41.77±8.60	96.95±0.31	**98.05±0.12**	**98.54±0.04**	
EL2N	36.13±8.15	94.90±0.61	97.45±0.26	98.27±0.01	
LCMat	66.89±4.74	95.96±0.10	97.47±0.13	98.02±0.10	
AdaCore	64.65±4.38	96.17±0.30	97.24±0.17	97.87±0.13	
CCS	56.84±7.70	96.77±0.11	97.88±0.14	98.28±0.10	98.60±0.03
Moderate DS	66.84±3.38	95.87±0.06	97.17±0.14	97.73±0.05	
Random	66.34±4.61	96.37±0.21	97.42±0.12	98.01±0.11	
BWS (Ours)	**71.42±3.54**	**97.05±0.23**	98.03±0.07	98.45±0.04	
Oracle window	73.81±2.00	97.16±0.25	98.06±0.09	98.45±0.04	

Table 19. Test accuracy of 20% label-noise CIFAR-10 dataset. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	10%	20%	30%	40%	100%
Forgetting	49.82±0.87	64.52±1.60	69.89±1.10	72.21±0.10	
EL2N	7.02±0.22	9.48±0.15	21.11±0.19	38.91±0.87	
LCMat	59.20±1.21	71.54±1.28	77.77±0.48	81.23±0.42	
AdaCore	10.96±0.10	10.85±0.26	39.14±0.65	58.51±0.20	
CCS	56.87±0.52	72.20±0.60	77.16±0.61	80.22±0.73	82.66±0.00
Moderate DS	**78.75±0.32**	**86.53±0.24**	**89.61±0.32**	**91.35±0.21**	
Random	64.24±1.35	75.45±0.67	79.58±0.70	81.99±0.27	
BWS (Ours)	78.74±0.56	84.77±0.28	88.06±0.03	89.79±0.08	
Oracle window	81.01±0.21	87.32±0.12	90.06±0.06	91.48±0.09	

Table 20. Test accuracy of 40% label-noise CIFAR-10 dataset. We highlight the highest values in bold and the second-highest values in underscore.

Selection ratio	10%	20%	30%	40%	100%
Forgetting	53.90±1.26	66.95±1.09	70.78±0.18	69.65±0.39	
EL2N	5.75±0.19	5.81±0.17	5.68±0.27	8.63±0.14	
LCMat	43.32±0.44	57.73±1.35	65.00±0.39	69.11±0.31	
AdaCore	9.76±0.48	10.03±0.09	10.42±0.32	10.73±0.21	
CCS	52.08±2.15	64.65±1.34	70.08±0.19	72.09±0.18	72.71±0.40
Moderate DS	77.69±0.97	**86.21±0.19**	**88.63±0.13**	83.42±0.16	
Random	50.33±0.40	60.21±1.28	65.01±0.60	67.68±0.65	
BWS (Ours)	**78.40±0.37**	85.42±0.08	87.76±0.17	**88.97±0.16**	
Oracle window	80.58±0.41	86.06±0.17	88.12±0.28	88.97±0.16	

Table 21. Comparison of window subsets of CIFAR-10 in terms of their 1) test accuracy, measured on models trained with the window subsets (top rows) and 2) accuracy of kernel ridge regression on the training dataset (bottom rows). The best performing windows align well between the two measures.

Ratio	Starting point	0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%	70%	75%	80%	85%	90%
10%	Test Acc	56.34	58.34	68.54	70.24	74.77	78.54	81.32	81.71	82.24	82.46	**82.67**	82.29	80.95	80.53	79.75	77.99	77.41	74.88	71.09
	Regression Acc	56.93	59.32	61.41	63.94	65.01	65.93	67.03	67.25	67.55	67.81	67.74	**67.81**	67.73	67.55	67.27	66.91	66.65	65.98	65.10
20%	Test Acc	79.08	83.39	86.03	87.79	88.33	**89.06**	88.74	88.42	88.06	87.23	86.86	86.12	85.37	84.29	83.10	82.09	80.42	-	-
	Regression Acc	76.27	77.07	77.85	78.98	79.41	79.69	**79.83**	79.79	79.61	79.49	79.29	79.06	78.86	78.63	78.35	78.16	77.81	-	-
30%	Test Acc	89.45	91.14	91.77	**91.80**	91.67	90.94	90.68	89.97	89.47	88.92	88.13	87.47	86.69	85.78	84.47	-	-	-	-
	Regression Acc	83.81	84.23	84.42	**84.49**	84.47	84.34	84.20	84.00	83.85	83.70	83.50	83.33	83.17	83.05	82.86	-	-	-	-
40%	Test Acc	93.08	**93.59**	93.39	93.00	92.46	91.63	91.11	90.54	89.88	89.02	88.46	87.76	86.96	-	-	-	-	-	-
	Regression Acc	87.42	**87.48**	87.39	87.30	87.19	87.04	86.88	86.71	86.58	86.42	86.29	86.18	86.02	-	-	-	-	-	-

Table 22. Comparison of window subsets of CIFAR-100 in terms of their 1) test accuracy, measured on models trained with the window subsets (top rows) and 2) accuracy of kernel ridge regression on the training dataset (bottom rows). The best performing windows align well between the two measures.

Ratio	Starting point	0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%	70%	75%	80%	85%	90%
10%	Test Acc	35.64	34.81	28.31	30.53	33.22	33.07	32.12	38.03	34.69	37.98	38.36	43.06	42.24	41.73	43.60	44.11	**45.16**	42.77	42.70
	Regression Acc	11.12	11.23	11.02	10.89	10.96	11.28	12.11	12.34	12.50	12.84	12.88	13.12	13.48	13.60	13.99	**14.24**	14.16	14.02	14.06
20%	Test Acc	48.78	52.02	49.43	53.03	54.39	54.99	56.60	56.51	57.08	56.83	58.70	**58.91**	58.30	56.26	56.34	56.71	52.96	-	-
	Regression Acc	25.62	25.58	25.78	25.76	26.48	27.00	27.36	27.59	27.67	27.85	28.17	28.29	**28.51**	28.44	28.23	27.93	27.30	-	-
30%	Test Acc	62.25	61.17	62.66	63.78	66.62	67.20	66.56	65.33	**67.51**	66.61	63.93	64.88	63.09	60.47	59.06	-	-	-	-
	Regression Acc	43.27	43.46	43.83	43.85	44.11	**44.37**	44.22	44.23	44.01	43.78	43.56	43.21	42.56	41.95	40.92	-	-	-	-
40%	Test Acc	70.50	69.91	72.17	70.78	**72.70**	72.11	69.79	71.38	69.06	68.87	67.83	66.25	63.98	-	-	-	-	-	-
	Regression Acc	54.05	54.36	**54.44**	54.30	53.96	53.67	53.25	52.74	52.23	51.56	50.72	49.73	48.64	-	-	-	-	-	-

Table 23. Comparison of window subsets of ImageNet in terms of their 1) test accuracy, measured on models trained with the window subsets (top rows) and 2) accuracy of kernel ridge regression on the training dataset (bottom rows).

Ratio	Starting point	0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%	70%	75%	80%	85%	90%
10%	Test Acc	46.78	46.34	46.49	46.89	45.50	47.07	45.82	48.07	**49.21**	48.00	48.62	47.76	46.73	46.67	46.64	45.03	43.43	40.08	32.68
	Regression Acc	35.05	35.45	35.71	35.84	35.74	35.72	35.89	35.99	36.03	36.06	36.05	36.04	36.19	36.23	**36.24**	36.17	35.90	35.01	32.43
20%	Test Acc	61.02	62.08	60.60	**62.62**	61.49	62.38	62.29	61.34	61.46	61.02	59.77	59.46	58.53	56.91	54.43	52.37	46.93	-	-
	Regression Acc	44.72	**44.91**	44.83	44.72	44.59	44.54	44.35	44.21	44.12	43.94	43.89	43.84	43.75	43.65	43.39	42.81	41.30	-	-
30%	Test Acc	67.28	**68.27**	67.89	67.90	68.00	67.81	67.85	66.87	66.91	65.68	64.53	62.77	61.66	58.88	55.25	-	-	-	-
	Regression Acc	**52.19**	52.14	51.93	51.70	51.52	51.26	51.04	50.83	50.65	50.46	50.30	50.14	49.86	49.39	48.36	-	-	-	-
40%	Test Acc	70.53	70.65	**71.35**	71.27	70.87	70.31	69.87	68.79	68.42	67.05	66.11	63.92	61.54	-	-	-	-	-	-
	Regression Acc	**53.48**	53.41	53.23	52.98	52.74	52.54	52.27	52.05	51.87	51.67	51.42	50.99	50.14	-	-	-	-	-	-

Table 24. Comparison of window subsets of CIFAR-10 dataset with 20% label noise in terms of their 1) test accuracy, measured on models trained with the window subsets (top rows) and 2) accuracy of kernel ridge regression on the training dataset (middle rows). We also report the noise portion with each window subset (bottom rows). The best window alignment between the two measures gets less accurate, compared to the case without label noise, since our method (regression) tends to choose more easier samples. However, such tendency also makes the choice of window subset mostly composed of clean-label samples.

Ratio	Starting point	0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%	70%	75%	80%	85%	90%
10%	Test Acc	6.98	8.93	13.80	31.90	58.00	68.77	75.21	78.29	79.18	79.04	79.85	79.99	81.01	80.86	79.72	79.53	78.74	77.81	75.71
	Regression Acc	51.54	55.95	58.10	59.52	60.42	61.68	63.86	64.94	66.11	67.58	68.74	69.64	69.73	70.09	70.20	70.43	70.54	70.13	69.73
	Noise Portion	92%	85%	69%	39%	15%	8%	5%	4%	4%	4%	3%	3%	3%	3%	3%	3%	3%	3%	3%
20%	Test Acc	9.56	17.79	35.29	63.55	78.99	84.46	86.52	87.07	87.32	87.17	86.85	86.47	86.32	85.36	84.77	84.08	82.82	-	-
	Regression Acc	73.28	75.17	76.06	77.03	78.38	78.66	79.05	79.63	80.01	80.47	80.67	80.67	80.66	80.63	80.70	80.60	80.35	-	-
	Noise Portion	80%	62%	42%	24%	10%	6%	5%	4%	4%	3%	3%	3%	3%	3%	3%	3%	3%	-	-
30%	Test Acc	20.90	38.99	62.51	79.74	87.82	89.80	90.06	89.83	89.34	88.94	88.46	88.06	87.66	86.93	86.04	-	-	-	-
	Regression Acc	80.43	81.39	82.33	83.20	83.45	83.84	84.10	84.30	84.36	84.55	84.47	84.60	84.39	84.41	84.16	-	-	-	-
	Noise Portion	59%	44%	30%	17%	8%	5%	4%	4%	3%	3%	3%	3%	3%	3%	3%	-	-	-	-
40%	Test Acc	38.21	59.90	77.70	87.11	90.70	91.48	91.32	90.75	90.10	89.79	89.04	88.84	88.16	-	-	-	-	-	-
	Regression Acc	86.19	86.50	86.63	86.87	86.96	87.11	87.21	87.16	87.29	87.39	87.25	87.25	87.16	-	-	-	-	-	-
	Noise Portion	45%	34%	23%	14%	7%	5%	4%	4%	3%	3%	3%	3%	3%	-	-	-	-	-	-

Table 25. Comparison of window subsets of CIFAR-10 dataset with 40% label noise in terms of their 1) test accuracy, measured on models trained with the window subsets (top rows) and 2) accuracy of kernel ridge regression on the training dataset (middle rows). We also report the noise portion in each window subset (bottom rows). The best window alignment between the two measures gets less accurate, compared to the case without label noise, since our method (regression) tends to choose more easier samples. However, such tendency also makes the choice of window subset mostly composed of clean-label samples.

Ratio	Starting point	0%	5%	10%	15%	20%	25%	30%	35%	40%	45%	50%	55%	60%	65%	70%	75%	80%	85%	90%
10%	Test Acc	6.28	6.12	6.82	7.82	7.60	9.81	21.05	45.22	62.11	73.62	77.57	78.94	80.58	79.96	79.59	78.40	77.92	77.07	75.51
	Regression Acc	1.87	1.88	1.47	2.16	3.13	12.03	25.36	46.69	56.92	61.44	63.84	65.21	65.17	65.18	65.45	65.63	65.11	64.73	64.03
	Noise Portion	92%	93%	92%	90%	88%	82%	61%	35%	21%	15%	13%	11%	10%	9%	9%	8%	7%	8%	9%
20%	Test Acc	5.61	6.62	6.84	8.17	13.36	28.17	52.85	72.56	82.01	85.09	86.06	85.59	85.42	84.86	83.92	83.01	82.05	-	-
	Regression Acc	0.89	0.98	1.35	3.32	14.45	39.13	60.53	69.82	73.12	74.90	75.85	75.86	75.93	75.56	75.27	74.93	74.35	-	-
	Noise Portion	92%	91%	90%	86%	74%	58%	41%	25%	17%	13%	11%	10%	9%	8%	8%	8%	8%	-	-
30%	Test Acc	5.46	6.71	10.28	17.60	35.26	56.16	72.82	83.20	87.32	88.12	87.76	87.61	86.86	86.17	85.23	-	-	-	-
	Regression Acc	0.48	0.98	4.05	18.44	51.29	71.18	79.48	82.93	84.46	85.28	85.45	85.38	85.03	84.82	84.45	-	-	-	-
	Noise Portion	90%	88%	80%	69%	56%	44%	32%	21%	14%	12%	11%	9%	9%	8%	8%	-	-	-	-
40%	Test Acc	8.50	12.79	22.46	39.42	58.61	72.52	81.96	87.17	88.90	88.97	88.63	88.11	87.28	-	-	-	-	-	-
	Regression Acc	2.85	11.02	30.74	60.30	79.54	86.17	88.11	88.82	89.34	89.46	89.36	89.16	88.91	-	-	-	-	-	-
	Noise Portion	83%	75%	65%	55%	46%	36%	26%	18%	13%	11%	10%	9%	9%	-	-	-	-	-	-