

Partial Forward Blocking: A Novel Data Pruning Paradigm for Lossless Training Acceleration

Dongyue Wu, Zilin Guo, Jialong Zuo, Nong Sang, Changxin Gao*

National Key Laboratory of Multispectral Information Intelligent Processing Technology,
School of Artificial Intelligence and Automation, Huazhong University of Science and Technology
{dongyue_wu, zilin_guo, jlongzuo, nsang, cgao}@hust.edu.cn

Abstract

The ever-growing size of training datasets enhances the generalization capability of modern machine learning models but also incurs exorbitant computational costs. Existing data pruning approaches aim to accelerate training by removing those less important samples. However, they often rely on gradients or proxy models, leading to prohibitive additional costs of gradient back-propagation and proxy model training. In this paper, we propose Partial Forward Blocking (PFB), a novel framework for lossless training acceleration. The efficiency of PFB stems from its unique adaptive pruning pipeline: sample importance is assessed based on features extracted from the shallow layers of the target model. Less important samples are then pruned, allowing only the retained ones to proceed with the subsequent forward pass and loss back-propagation. This mechanism significantly reduces the computational overhead of deep-layer forward passes and back-propagation for pruned samples, while also eliminating the need for auxiliary backward computations and proxy model training. Moreover, PFB introduces probability density as an indicator of sample importance. Combined with an adaptive distribution estimation module, our method dynamically prioritizes relatively rare samples, aligning with the constantly evolving training state. Extensive experiments demonstrate the significant superiority of PFB in performance and speed. On ImageNet, PFB achieves a 0.5% accuracy improvement and 33% training time reduction with 40% data pruned.

1. Introduction

The availability of large-scale datasets[1, 4, 22, 31, 39] in recent years has propelled significant advancements in deep learning. Many powerful vision models[22, 36, 38] rely on vast amounts of training data to achieve robust gener-

*Corresponding author.

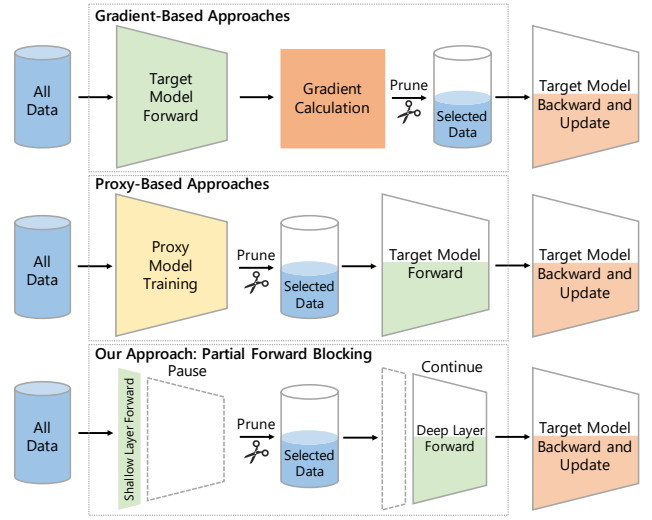


Figure 1. Computation cost comparison between Partial Forward Blocking (PFB) and existing approaches. The half blanks in **Forward** and **Update** indicate that only the selected samples participate in the network’s forward inference, back-propagation, and updates. The proposed PFB strategy avoids the additional cost of gradient calculation and proxy model training by pruning based on features of the shallow network and blocking the subsequent deep network forward pass of the pruned samples.

alization. However, behind the unprecedented generalization performance lies a critical challenge: the enormous volume of training samples substantially increases the computational cost of training. Thus, reducing the training cost has become a key and pressing concern.

Replacing large datasets with a compact subset offers an intuitive and effective solution. Dataset distillation[3, 32, 47, 51, 52] aims to synthesize a small yet highly informative set of samples by leveraging information from the original dataset. However, the synthetic data often deviates significantly from real-world samples. In contrast, data pruning[8, 33, 44, 49] and online batch selection[17, 26, 28] directly selects a subset of real samples, ensuring authenticity. Yet, these existing methods typically rely on additional

proxy models[8, 13, 28] or computationally expensive metrics, such as gradients[17, 33, 45, 49], to assess sample importance. The computational burden introduced by proxy models, along with the cost of computing losses and back-propagating gradients, becomes increasingly prohibitive as dataset size grows, thereby undermining their efficiency.

This raises a critical question: *Can we efficiently select training samples at a lower cost while preserving the strong generalization of target models?* We believe an effective solution to this problem should possess the following characteristics: 1) **Efficiency**: Its computational overhead of importance evaluation should be minimal, making it feasible for large-scale datasets. 2) **Diversity**: The selected samples should span a broad and informative distribution to ensure a strong generalization and powerful performance. 3) **Adaptability**: The method should dynamically adjust to the evolving training state of the model.

This paper proposes a novel paradigm that significantly reduces training costs while preserving model performance to achieve these goals. We introduce the Partial Forward Blocking (PFB) strategy, which prunes samples in a batch-wise manner at the early stage of the forward pass to eliminate unnecessary computation as shown in Fig. 1. First, the target network is partitioned into a shallow sub-network and a deeper subsequent one. Importance scores are computed based on the features from the shallow sub-network. Samples with low importance are discarded immediately, blocking their forward pass through the deeper layers, thereby reducing computation costs. Meanwhile, the retained samples reuse their extracted features to continue forward propagation, further improving efficiency. Compared with gradient-based and proxy-based works, the PFB strategy avoids the substantial overhead of gradient back-propagation and proxy model training for importance scoring and significantly reduces the forward cost of the pruned samples.

To ensure the diversity of retained samples, we introduce a new importance metric, which leverages probability density as an indicator of sample redundancy: samples with high probability density are common and less informative, making them less important, while those with low probability density are rare and valuable for generalization. This criterion ensures the diversity of the retained subset by prioritizing rare samples and preventing excessive concentration in densely populated feature regions. Additionally, an Adaptive Distribution Estimation (ADE) module continuously updates its estimated feature distribution based on retained samples, enabling PFB to dynamically adjust to the evolving training state. By adaptively favoring rare and informative samples, PFB ensures that the retained subset remains well-aligned with the model’s learning needs. Extensive experiments demonstrate that PFB achieves superior training efficiency with minimal computational overhead while maintaining competitive performance. For ex-

ample, PFB prunes 40% training samples of ImageNet-1k but still achieve a 0.5% accuracy improvement and 33.2% training time reduction over the full data baseline. In summary, our contributions are as follows:

1. We propose an efficient training acceleration pipeline that conducts pruning in a batch-wise manner at the early stage of forward and blocks the subsequent forward pass of pruned samples.
2. To ensure the diversity of selected samples, we propose utilizing the probability density of the feature distribution to measure importance.
3. We introduce a distribution estimation module that adaptively tracks the distribution shift during training.

2. Related Work

Extracting a small amount of valuable samples from large datasets is a common objective in many other tasks, such as active learning[2, 41], noisy learning[30], curriculum learning[50], and data distillation[32, 47]. Since our approach focuses on accelerating training using real data, we primarily introduce data pruning and online batch selection.

2.1. Data Pruning

Data Pruning aims at selecting a small subset of training samples that can achieve comparable results as the original dataset to accelerate model training. These methods can be divided into two types: static and dynamic data pruning.

Static methods only prune the dataset once. Thus, the performance of target models trained on the pruned datasets demonstrates the effectiveness of static methods. The pioneering work *Herding* [7] tends to keep samples that are close to class centers. EL2N [33] quantifies the sample-wise learning difficulty by averaging the norm of error of a set of networks. Similarly, GraNd [33] adopts the expectation of the gradient norm as an importance indicator. *Dataset Pruning* [49] and MoSo [45] prune the samples which do the least damage to the empirical risk based on back-propagated gradients. SVP [8] and YOCO [16] both select samples according to the entropy and error of the prediction by pre-trained proxy models, respectively. On the other hand, *Forgetting* [46] measures the learning difficulty via the forgetting times during training, namely the change of prediction from correct to incorrect.

Recently, dynamic pruning methods that periodically select samples during training have been proposed. Raju et al.[37] first introduced a strategy where the pruning decision for a given sample can change across different pruning cycles. Their proposed methods, UCB and ϵ -greedy, dynamically select different samples for training based on prediction uncertainty at each pruning stage. Subsequently, Dyn-Unc [15] adopts a similar pipeline and measures uncertainty in a sliding window of successive training epochs as a pruning metric. InfoBatch [35] mitigates distribution

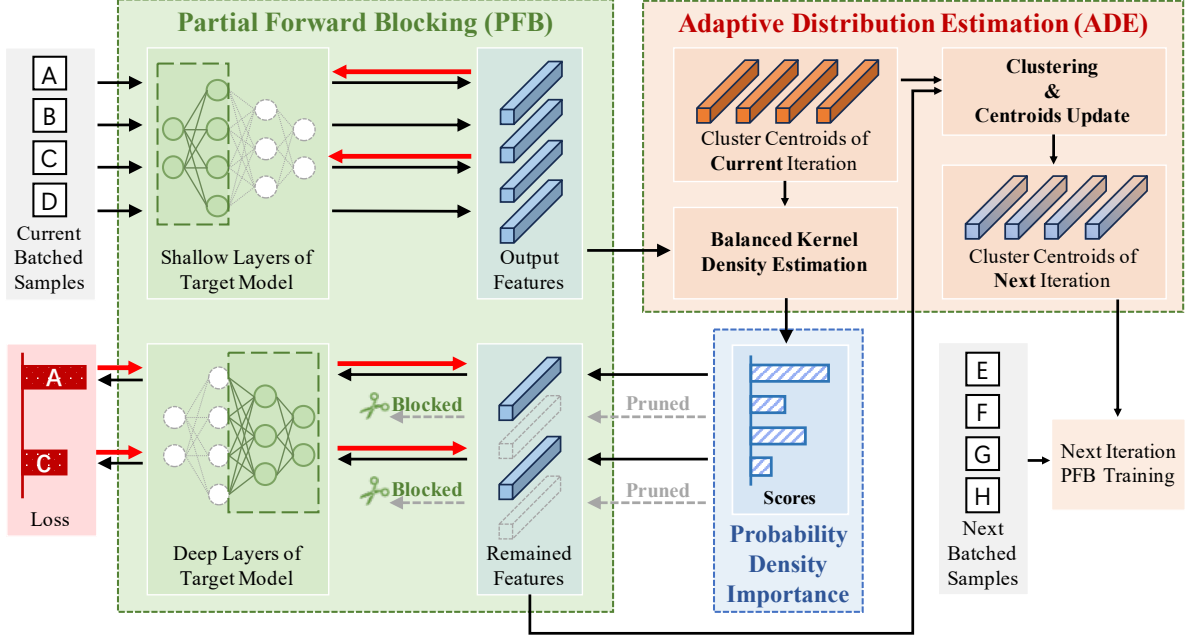


Figure 2. Illustration of the proposed Partial Forward Blocking pipeline. The *black* and *red* arrows denote forward and backward procedures, respectively. We use A-H to denote individual samples. We adopt an online paradigm which prune samples and save the training cost by block the forward and backward pass of the pruned samples in a batch-wise manner. We introduce the Adaptive Distribution Estimation module to obtain probability density, based on which the Probability Density Importance is computed.

shifts by removing a proportion of low-loss samples while assigning higher weights to the remaining ones in an epoch-wise manner. Despite their getting rid of the burden of both gradient calculation and proxy model training, they still require a full forward pass on all samples in the dataset for sample-wise importance assessment and pruning.

2.2. Online Batch Selection

Online batch selection [18, 26] can be regarded as a special type of data pruning, which is more flexible as it selects samples within each batch at every training iteration. Despite this flexibility, many methods still rely on additional surrogate models and expensive gradients, leading to limited efficiency. For example, RHO-Loss [28] requires training a surrogate model on a holdout dataset with a certain number of samples, while Deng et al. [13] use a pre-trained zero-shot predictor as their surrogate model. On the other hand, while gradient-based online batch selection methods [17, 19] save the expense of proxy training, they incur additional computational costs due to the need for loss back-propagation to obtain gradients.

2.3. Comparison with Existing Methods

Compared to proxy-model-based methods, our approach eliminates the training cost associated with these models. Unlike gradient-based methods, our method avoids the costly and additional backward propagation for gradi-

ent calculation. Even compared to methods like SB[18], UCB[37] and InfoBatch[35] that mitigate both issues, our approach requires only a small portion of shallow network computations for pruned samples during the forward pass, saving the substantial computational cost in deeper layers. On the contrary, these methods still perform the full, expensive forward pass on pruned samples.

3. Methodology

3.1. Problem Definition

Given a dataset Ψ , our task is to train a deep network Net_θ with parameters θ . At the t -th iteration during training, a batch of samples $B^t = \{z_i^t\}_{i=1}^{N_B}$ is randomly sampled from Ψ . The samples with the least importance scores in the current t -th batch B^t are pruned with a prune ratio $p \in (0, 1)$, which can be formulated as optimization problem:

$$\min_{m_i^t} \sum_{i=1}^{N_B} m_i^t \mathcal{I}(z_i^t), \quad s.t. \sum_{i=1}^{N_B} m_i^t = p N_B, \quad (1)$$

where $\mathcal{I}(z_i^t)$ denotes the importance score of z_i^t , and $m_i^t \in \{0, 1\}$ signifies its pruning decision, where $m_i^t = 1$ indicates z_i^t is pruned. Hence, by solving Eq. (1) we can get the pruning decisions $M^t = \{m_i^t\}_{i=1}^{N_B}$ and conduct a subset batch $S^t = \{z_i^t | m_i^t = 0\}$ with a smaller size $|S^t| = (1 - p)N_B$. For each training iteration, the target

model is only trained on S^t . Compared with the vanilla training procedure, the training cost on the subset of the pruned samples $B^t \setminus S^t$ is saved. Our goal is to maintain strong performance and speed up training simultaneously.

3.2. Overall Pipeline of Partial Forward Blocking

The training cost of pruning methods consists of three components: Forward, Backward & Parameter Update (B&PU), and Scoring. Since most methods perform B&PU only on retained samples, its cost can not be further reduced. Thus, *how to effectively reduce the expense of Forward and Scoring* is the key to training acceleration.

Let us begin with analyzing the Forward and Scoring costs of existing methods. Gradient-based approaches rely on loss gradients to compute importance scores, requiring a full forward pass (for Forward) and loss backpropagation (for Scoring) on all samples before pruning. Some methods, such as InfoBatch [35], score samples directly based on loss values, reducing Forward and Scoring overhead to a single forward pass on all samples. However, they fail to save the substantial costs of forward pass on pruned samples. In contrast, proxy-based methods use lightweight models for early pruning, ensuring that only retained samples undergo a forward pass in the target model. However, these approaches require training proxy models for Scoring, which entails a computational cost comparable to training the target model itself, making them impractically expensive.

To overcome these limitations, we propose Partial Forward Blocking (PFB), which computes importance scores and prunes samples in the early stage of forward propagation. Specifically, we derive importance scores from shallow network features, allowing retained samples to reuse these features for the remainder of their forward pass. By pruning early, our method eliminates unnecessary computation on pruned samples in deeper layers, significantly reducing Forward costs compared to gradient-based methods. Furthermore, for Scoring, we only incur the minor computational cost of shallow layers (e.g., stage-1) without requiring proxy model training. As a result, the total computational cost of Forward and Scoring is limited to a full forward pass on retained samples and a shallow forward pass on pruned ones. This makes our method significantly more efficient than prior approaches. The details are outlined in Algorithm 1. We use $\mathbb{1}[\cdot]$ to denote the indicator function.

3.3. Probability Density Importance Criterion

It is commonly accepted that strong generalization requires diversified training samples. *How, then, can we design a per-sample importance criterion to ensure that pruning retains this diversity?* A natural approach is to eliminate highly redundant samples. If the entire dataset is available, redundancy can be estimated by exhaustively computing pairwise feature similarities. However, for ultra-large

Algorithm 1: Partial Forward Blocking Pipeline

```

1 Input: Target model  $Net(\cdot)$ , which can be divided
   into shallow sub-model  $Net^{sh}(\cdot)$  and a deep
   sub-model  $Net^{dp}(\cdot)$ , its initial parameters  $\theta^0$ ,
   pruning ratio  $p$ .
2 Output: The updated parameters  $\theta^T$ .
3 for  $t \in [1, T]$  do
4    $B^t \leftarrow \text{Sampler}(\mathcal{D})$ 
   #Get feature map  $\mathbf{f}_i^t \in \mathcal{R}^{H \times W \times D_{org}}$ 
5    $F^t \leftarrow \{\mathbf{f}_i^t = Net_{\theta^{t-1}}^{sh}(z_i^t) \mid z_i^t \in B^t\}$ 
   #Pooled representation  $\mathbf{x}_i^t \in \mathcal{R}^{1 \times D}$ 
6    $F_{rp}^t \leftarrow \{\mathbf{x}_i^t = \text{Pool}(\mathbf{f}_i^t) \mid \mathbf{f}_i^t \in F^t\}$ 
7    $I^t \leftarrow \{\mathcal{I}(z_i^t) = \text{Imp}(\mathbf{x}_i^t) \mid \mathbf{x}_i^t \in F_{rp}^t\}$  #Eq.2-3
   #Prune less important samples
8    $\tau \leftarrow \text{Percentile}(I^t, p)$  #Get threshold
9    $M^t \leftarrow \{m_i^t = \mathbb{1}[\mathcal{I}(z_i^t) < \tau] \mid \mathcal{I}(z_i^t) \in I^t\}$ 
10   $S^t \leftarrow \{z_i^t \mid m_i^t = 0\}$ 
   #Block forward of the pruned ones
11   $L^t \leftarrow \{Loss(Net_{\theta^{t-1}}^{dp}(\mathbf{f}_i^t)) \mid z_i^t \in S^t\}$ 
12   $\theta^t \leftarrow \text{Update}(\theta^{t-1}, L^t)$ 
13   $t \leftarrow t + 1$ 
14 return  $\theta^T$ 

```

datasets, this brute-force method is impractical due to two major challenges. First, extracting and storing feature representations for all samples imposes significant storage overhead. Second, such a similarity calculation demands prohibitive time and computation.

To tackle this problem, we first reframe it in terms of data distribution. Diversity can be preserved by maintaining a broad distribution and preventing excessive concentration of retained samples. In the feature distribution, samples in dense regions have many similar counterparts, making them highly redundant and less important. Even if a significant portion of these samples is removed, their retained counterparts can still preserve the essential information. Conversely, samples in sparse regions are relatively rare, and removing them would drastically reduce the likelihood of selecting similar samples, losing valuable learning opportunities. Thus, these samples exhibit low redundancy and high importance. Therefore, estimating the number or proportion of similar counterparts for each sample provides an indirect measure of its importance.

Following this analysis, we propose leveraging the probability density of a sample’s feature distribution as an importance metric. Probability density reflects the relative likelihood of a sample with a given feature representation occurring in the dataset. High probability density suggests a greater prevalence of similar samples, indicating substantial redundancy and diminished importance. Therefore, for

a sample z_i , its probability density importance $\mathcal{I}(z_i)$ is defined as follows:

$$\mathcal{I}(z_i^t) = \frac{1}{f_{\mathbf{X}}(\mathbf{x}_i^t) + r}, \quad (2)$$

where \mathbf{X} denotes the vector-valued random variable (feature representation) that conforms to the probability distribution of training samples, $f_{\mathbf{X}}(\cdot)$ is the probability density function (PDF), and r is set as follows:

$$r = \alpha \cdot \max_{z_i \in B} f_{\mathbf{X}}(\mathbf{x}_i^t). \quad (3)$$

where $\alpha \sim U(0, b)$ is randomly sampled from the uniform distribution, and the upper bound b is set as 0.01. The ablation study of this random term can be found in Sec. 4.3. We add the this term (r) in Eq. (2) to complement some randomness of the pruning process to further ensure the diversity of training data.

3.4. Adaptive Distribution Estimation

In Sec. 3.3, we propose to quantify the importance of all samples in each batch using their probability density. Nevertheless, *how can we compute the PDF adaptively to track the distribution shift caused by parameter updating while maintaining a negligible computational cost?*

To address this challenge, we propose the Adaptive Distribution Estimation (ADE), which uses Kernel Density Estimation (KDE) to estimate the probability density. By clustering samples in each batch and updating centroids to represent previously retained samples, we reduce the number of kernels in KDE computation. We also track the number of samples each centroid represents and apply weight coefficients to balance kernel contributions. This approach enables efficient, adaptive distribution tracking with minimal computational overhead.

Balanced Kernel Density Estimation. For each sample z_i^t in the current batch B^t of t -th training iteration, we conduct spatial average pooling on its original feature map $\mathbf{F}_i^t \in \mathcal{R}^{H \times W \times D_{org}}$ to a $1 \times D_{org}$ vector, and then further down-sample on the channel dimension to a $1 \times D$ shaped vector \mathbf{x}_i^t as the representation of z_i^t , in order to improve the computation efficiency. Then, kernel functions are applied to the current set of centroids $C^t = \{\mathbf{c}_j^t\}_{j=1}^{N_C}$ to estimate the probability density of z_i^t using its representation \mathbf{x}_i^t :

$$\hat{f}_{\mathbf{X}}(\mathbf{x}_i^t) = \sum_{j=1}^{N_C} \frac{w_j^t}{N_C} K_{\mathbf{H}}(\mathbf{x}_i^t - \mathbf{c}_j^t), \quad (4)$$

where $\mathbf{c}_j^t \in \mathcal{R}^{1 \times D}$ is the feature of the j -th centroid among all N_C ones, and $K_{\mathbf{H}}(\cdot)$ is the scaled kernel function. Please note that we introduce w_j^t to balance different

kernels, which will be explained later in Eq. (9). For simplicity, we use the standard multivariate normal kernel:

$$K_{\mathbf{H}}(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\mathbf{H}|^{1/2}} e^{-\frac{1}{2} \mathbf{x}^\top \mathbf{H}^{-1} \mathbf{x}}, \quad (5)$$

where $\mathbf{H} \in \mathcal{R}^{D \times D}$ is the bandwidth serving as the covariance matrix and can be selected using various bandwidth selection methods [40, 43]. Following Silverman's rule of thumb[43], \mathbf{H} is set as a diagonal matrix:

$$\mathbf{H} = \text{diag}(h_1, h_2, \dots, h_D),$$

$$\sqrt{h_d} = \left(\frac{4}{(D+2)N_C} \right)^{\frac{1}{D+4}} \sigma_d, \quad (6)$$

where the σ_d is the standard deviation of all centroids along the d -th dimension. Thanks to feature map pooling and the small number of centroids, the KDE process is efficient, as evidenced by results in Tab. 3. Based on Eqs. (4) and (5), we can compute $\mathcal{I}(z_i^t)$ for each z_i^t according to Eq. (2). Following the objective in Eq. (1), we can conduct the small batch S^t consisting of all the retained samples.

Clustering & Centroids Update.

We update centroids and compute weights using only retained samples. Each sample is assigned to its most similar centroid via clustering, formulated as finding a partition $\mathcal{P}(S^t) = \{A_1^t, A_2^t, \dots, A_{N_C}^t\}$ of S^t , where each $z_i^t \in S^t$ belongs to a cluster (subset) A_j^t that minimizes the Euclidean distance to its centroid \mathbf{c}_j^t :

$$\arg \min_{\mathcal{P}(S^t)} \sum_{j=1}^{N_C} \sum_{z_i^t \in A_j^t} \|\mathbf{x}_i^t - \mathbf{c}_j^t\|^2. \quad (7)$$

Then, centroids are updated for KDE of the next iteration:

$$\mathbf{c}_j^{t+1} = \frac{1}{n_j^t} \left[\beta n_j^{t-1} \cdot \mathbf{c}_j^t + (1 - \beta) \sum_{z_i^t \in A_j^t} \mathbf{x}_i^t \right], \quad (8)$$

$$n_j^t = |A_j^t| + n_j^{t-1} = \sum_{iter=1}^t |A_j^{iter}|,$$

where $\beta=0.01$ is the coefficient of EMA, n_j^{t-1} and n_j^t are the recorded number of all samples that were previously assigned to the j -th centroid until the last and current training iteration, respectively. These continuously evolving centroids entitle our method with strong adaptability to the shifting distribution. Moreover, the number of samples previously assigned to each centroid is also used to construct the weight w_j^t :

$$w_j^{t+1} = \frac{n_j^t}{t \cdot (1 - p) N_B}. \quad (9)$$

If many similar samples were retained in previous iterations, their assigned centroids gain higher weights, reducing the importance of future similar samples. With these weights, our method adaptively favors retaining rarer samples, preserving diversity in the selected subset.

4. Experiment

4.1. Experimental Setup

Datasets. Following the most common setting of the latest works[15, 17, 35], we conduct experiments to evaluate the performance of PFB on CIFAR-10[24], CIFAR-100[24], and ImageNet-1k[12] for image classification. CIFAR-10/100 datasets consist of 32×32 sized images that are divided into 10 and 100 classes, respectively. Each of them contains 50,000 images for training and 10,000 for testing. ImageNet-1k is a large-scale dataset containing 1,281,167 training images and 50,000 validation images that are categorized into 1,000 classes. We also evaluate the efficiency of PFB on the semantic segmentation datasets PASCAL VOC 2012 trainaug[6] and Cityscapes[10]. PASCAL VOC 2012 trainaug consists of 10,582 and 1,456 images for training and testing, respectively. The Cityscapes contains 2,975 fine annotated images with a resolution of $2,048 \times 1,024$ for training and another 500 images for validation.

Implementation details. For image classification, we employ ResNet-18[14] as a backbone on CIFAR-10/100 trained for 200 epochs. The batch size is set as 128. On ImageNet-1k, we evaluate the performance of PFB using ResNet-50 and Swin-T[25] to evaluate the generalization across different network architectures. The batch size of both ResNet-50 and Swin-T is set as 1024. Other detailed settings can be found in our supplementary materials.

4.2. Comparisons with SOTA Methods

We compare our method with existing state-of-the-art data pruning and online batch selection methods on CIFAR-10/100 and ImageNet-1k. We define the pruning p as the proportion of pruned samples to all samples. For methods that prune the whole training dataset, like most data pruning methods[33, 35], the pruning ratio is set as $p = \frac{|\mathcal{D}|}{|\mathcal{D}_{pruned}|}$. As for methods (including ours) that adopt batch-wise pruning paradigm[17, 18], the pruning ratio is set as $p = \frac{N_S}{N_B}$. Note that InfoBatch[35] adopts a different pruning rate calculation strategy: it first selects only part of the samples with the lower loss as well-learned and applies pruning only within this subset, resulting in an actual pruning rate that is around only half of the reported value. To distinguish this, we denote the original InfoBatch with ‘*’ and our reproduction with ‘†’, which considers 95% of the samples as pruning candidates. We also introduce single-shot random pruning (Random) as a baseline for comparison. Other reproduced results following the same experimental setup of

Dataset	CIFAR-10			CIFAR-100		
Pruning Ratio	30%	50%	70%	30%	50%	70%
Random	94.6	93.3	90.2	73.8	72.1	69.7
Herding[7]	92.2	88.0	80.1	73.1	71.8	69.6
Influence[23]	93.1	91.3	88.3	74.4	72.0	68.9
K-Center[42]	94.7	93.9	90.9	74.1	72.2	70.2
SVP[8]	95.0	94.5	90.3	74.2	72.3	69.8
Craig[29]	94.8	93.3	88.4	74.4	71.9	69.7
SB-12hours†[18]	95.5	95.1	93.2	-	-	-
GraNd[33]	95.3	94.6	91.2	74.6	71.4	68.8
Glister[21]	95.2	94.0	90.9	74.6	73.2	70.4
ϵ -greedy[37]	95.2	94.9	94.1	76.4	74.8	-
UCB[37]	95.3	94.7	93.9	77.3	75.3	-
Forgetting[46]	94.7	94.1	91.7	75.3	73.1	69.9
EL2N[33]	95.3	95.1	91.9	77.2	72.1	-
Traning Loss[20]	-	-	94.6	-	-	72.6
AUM[34]	95.1	95.3	91.4	76.9	67.4	30.6
Moderate[48]	93.7	92.6	90.6	74.3	68.3	57.8
Grad Norm IS[19]	-	-	94.4	-	-	73.2
Dataset Pruning[49]	94.9	93.8	90.8	77.2	73.1	-
CCS[53]	95.4	95.0	93.0	77.1	74.5	68.9
MoSo†[45]	-	-	-	76.7	72.3	65.8
InfoBatch*[35]	95.6	95.1	94.7	78.2	78.1	76.5
DivBS†[17]	95.4	95.2	95.1	78.5	78.2	77.2
PFB(Ours)	95.9	95.5	95.2	79.1	78.8	77.9
	$\uparrow 0.3$	$\downarrow 0.1$	$\downarrow 0.4$	$\uparrow 0.9$	$\uparrow 0.6$	$\downarrow 0.3$
Full Data	95.6 \pm 0.1			78.2 \pm 0.1		

Table 1. Comparison to state-of-the-art methods on CIFAR-10/100 using ResNet-18. The proposed PFB achieves performance improvements when 30% samples are pruned. Under 50% pruning ratio, PFB still achieves nearly lossless results. * denotes different prune ratio settings. † denotes our reproduction.

PFB are also denoted with ‘†’.

Performance Comparisons. Top-1 accuracy on CIFAR-10/100 and ImageNet-1K is reported in Tab. 1 and Tab. 2, respectively. The results demonstrate that PFB consistently outperforms existing methods across all three datasets and various pruning ratios. Notably, at a 30% pruning ratio, PFB achieves better-than-lossless training, with ResNet models improving by 0.3%, 0.9%, and 0.6% over full-data training on CIFAR-10, CIFAR-100, and ImageNet-1K, respectively. These results strongly validate the diversity of the retained training samples and the enhanced generalization of the trained models. We attribute this improvement to both the proposed importance criterion and the Adaptive Distribution Estimation module.

Efficiency Comparisons on ImageNet-1k. The training efficiency of PFB is also compared with both classic methods (EL2N and UCB) and the latest methods (InfoBatch and DivBS). The time cost of ResNet-50 trained on ImageNet-1k for 90 epochs is reported in Tab. 3. EL2N and DivBS

Pruning Ratio		30%	50%	70%
ResNet-50	Random	72.2 _{↓4.2}	69.1 _{↓7.3}	65.9 _{↓10.5}
	Herdin[7]	73.5 _{↓2.9}	69.3 _{↓7.1}	65.1 _{↓11.3}
	Forgetting[46]	74.8 _{↓1.6}	72.0 _{↓4.4}	67.8 _{↓8.6}
	EL2N[33]	74.3 _{↓2.1}	68.5 _{↓7.9}	54.8 _{↓21.6}
	Moderate[48]	75.2 _{↓1.2}	72.2 _{↓4.2}	67.7 _{↓8.7}
	MoSo [†] [45]	76.5 _{↑0.1}	73.5 _{↓2.9}	70.0 _{↓6.4}
	InfoBatch [†] [35]	76.5 _{↑0.1}	75.8 _{↓0.6}	74.9 _{↓1.5}
	PFB(Ours)	77.0_{↑0.6}	76.1_{↓0.3}	75.3_{↓1.1}
Full Data		76.4 _{±0.2}		
Pruning Ratio		30%	40%	50%
Swin-T	Random	77.2 _{↓2.4}	75.9 _{↓3.7}	74.5 _{↓5.1}
	Forgetting[46]	78.3 _{↓1.3}	77.6 _{↓2.0}	74.3 _{↓5.3}
	EL2N[33]	78.2 _{↓1.4}	75.9 _{↓3.7}	71.1 _{↓8.5}
	SVP[8]	76.6 _{↓3.0}	74.9 _{↓4.7}	72.7 _{↓6.9}
	Moderate[48]	77.1 _{↓2.5}	75.9 _{↓3.7}	75.0 _{↓4.6}
	InfoBatch [†] [48]	78.6 _{↓1.0}	78.2 _{↓1.4}	77.5 _{↓2.1}
	Dyn-Unc[15]	79.1 _{↓0.5}	78.5 _{↓1.1}	77.6 _{↓2.0}
	PFB(Ours)	79.6_{↑0.0}	79.2_{↓0.4}	78.2_{↓1.4}
Full Data		79.6 _{±0.1}		

Table 2. Performance Comparison on ImageNet-1k. ResNet-50 and Swin-T are trained from scratch for 90 and 300 epochs, respectively. The results demonstrate the effectiveness of PFB on both CNNs and Transformers.

are representative proxy-based and gradient-based methods, respectively. As a result, they struggle to achieve satisfactory speedup on large-scale datasets like ImageNet due to their inherent computational overhead. In contrast, UCB and InfoBatch rank sample importance based on prediction uncertainty and loss values, effectively mitigating the high cost associated with proxy models and backpropagation in the previous approaches. Despite this, our proposed PFB mechanism further reduces training overhead by additionally skipping part of the forward pass for pruned samples, leading to even greater efficiency gains.

Efficiency Comparisons on Segmentation Datasets. To better demonstrate the acceleration advantage of PFB in skipping part of the forward pass, we further conduct experiments on a more computationally intensive semantic segmentation task, with results presented in Tab. 4. We apply InfoBatch, DivBS, and PFB to DeepLabV3-ResNet50[5], ensuring all methods undergo the same number of training iterations. Since semantic segmentation is a dense prediction task that requires per-pixel supervision, its forward inference and loss computation demand significantly more computational resources and time compared to classification tasks. Consequently, PFB achieves a more pronounced efficiency gain over InfoBatch and DivBS by skipping the forward pass through deep layers. Furthermore, as PFB

does not rely on gradient-based importance estimation, it exhibits a greater advantage over DivBS on Cityscapes than VOC, where images are larger and computational demands are higher. Experimental results show that with the PFB mechanism, PFB achieves a threefold reduction in training time compared to DivBS on Cityscapes.

4.3. Ablation Study

Location to Block. We first investigate the impact of the blocking location on performance. The results in Fig. 3(a) show that For both ResNet-50 and Swin-T, PFB performs well when blocking after stage-1 layers. When PFB is applied directly at the stem layers (stage-0) of ResNet-50, using the corresponding features as sample representations, we observe a significant drop in performance. We conjecture that this is due to the relatively noisy and less informative nature of shallow features in ResNet, which may not sufficiently capture the semantic distinctions and relationships between samples. To balance performance and efficiency, we therefore apply PFB at stage-2 for ResNet and stage-1 for Swin.

Batch Size. As the recorded centroids and the number of samples assigned to it is related to the number of samples in the batch, we conduct experiments to verify the robustness of PFB to different batch size settings on ImageNet-1k. Results in Fig. 3(d) reveal that the trend of PFB is consistent with that of full-data training. This phenomenon demonstrates that PFB is not sensitive to various batch sizes.

Random Term of Importance. The random term r in our proposed importance metric (Eq. (2)) enhances the diversity of retained samples, ensuring those with plenty of similar counterparts can sometimes be retained as well. We conducted five repeated experiments on the upper bound b , which controls the scale of r on CIFAR-100. We report the results in Fig. 3(b) with blue shading indicating the range between the maximum and minimum values. The results demonstrate that the accuracy remains stable when the upper bound b of random coefficient α is within [0.001, 1]. However, when α is excessively large, the random term dominates probability density in (Eq. (2)), leading to an excessive retention of randomly selected samples. Thus, the performance drops significantly with large fluctuations when b is set as 10.

Effectiveness on High Pruning Ratio. To explore the effectiveness under extreme conditions, we conduct ablation studies on CIFAR-100 with very large pruning ratios(0.7-0.9). As shown in Fig. 3(e), PFB still outperforms other powerful methods, including both gradient-based ones[17, 19] and those relying on proxy-models[8].

Coefficients in Adaptive Distribution Estimation. To find the best coefficients for probability density estimation, we compare the accuracy and the fluctuation under different settings in Fig. 3(c) and (f). The results show that

Method	Year	Pruning Freq.	Top-1 Acc(%)	Training(h)	Overhead(h)	Total(n*h)	Reduction
Full Data	-	-	76.4	13.9	-	55.6	-
EL2N [†] [33]	2018	Single-shot	71.5 _{↓4.9}	10.1	>14	>96	>72.7%↑
UCB [†] [37]	2021	Epoch	75.8 _{↓0.6}	10.1	0.08	40.8	26.7%↓
InfoBatch [†] [35]	2023	Epoch	76.5 _{↑0.1}	10.1	0.07	40.7	26.8%↓
DivBS [†] [17]	2024	Batch	76.4 _{↑0.0}	11.2	0.72	47.6	14.4%↓
PFB(ours)	-	Batch	76.9_{↑0.5}	9.2	0.06	37.1	33.2%↓

Table 3. Time cost comparison on ImageNet-1k using ResNet-50 under 40% pruning ratio. PFB saves the overall cost by more than 30% but still obtains a 0.5% accuracy gain. Results are collected on a 4-RTX 4090 GPU server. Automatic Mixed Precision[27] is adopted during training for all methods. ‘Training’ and ‘Overheads’ denote the wall clock time spent on network training and the additional time brought in by pruning methods. ‘Total(n*h)’ is the total node hour with n=4.

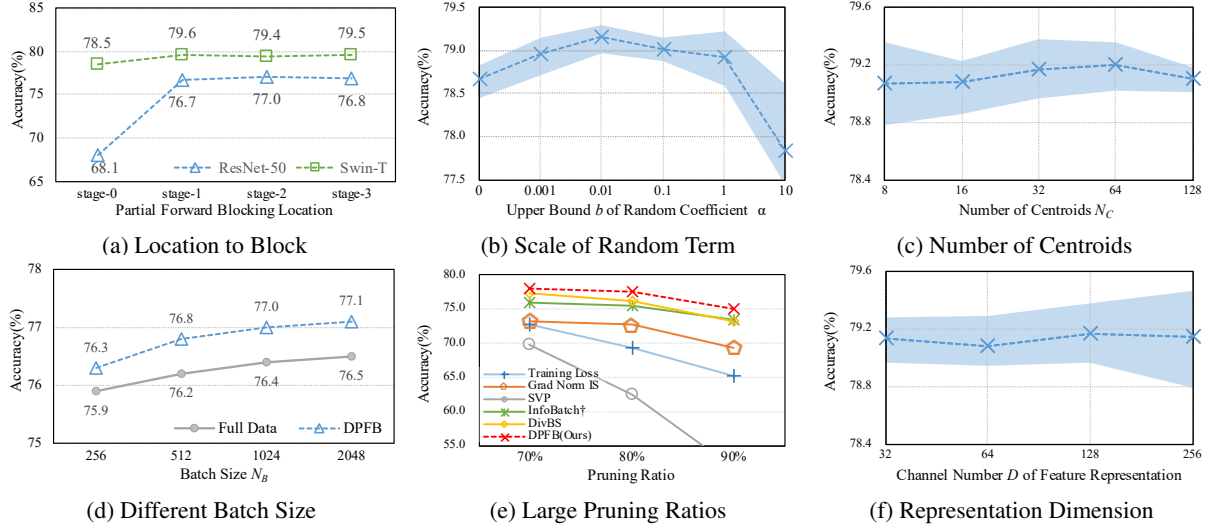


Figure 3. Ablation studies of PFB on (a) the blocking location of PFB strategy on ImageNet-1k, (b) the upper bound of the uniformly sampled coefficient of the random term, (c) the number of centroids for KDE, (d) the size of batch on ImageNet-1k, (e) performance under large pruning ratios, and (f) the channel number of feature representation. We use the shading to show the maximum and minimum of five independent runs. Unless otherwise specified, experiments are conducted on CIFAR-100.

Dataset	Method	mIoU	Time Saved
Cityscapes 30% Pruned	Full Data	80.4	-
	DivBS [†]	79.0 _{↓1.4}	7.8%↓
	InfoBatch [†]	79.2 _{↓1.2}	15.8%↓
	PFB(Ours)	79.8_{↓0.6}	24.1%↓
VOC 2012 70% Pruned	Full Data	77.9	-
	DivBS [†]	72.7 _{↓5.2}	42.6%↓
	InfoBatch [†]	72.4 _{↓5.5}	49.1%↓
	PFB(Ours)	73.4_{↓4.5}	56.7%↓

Table 4. Results comparison with other pruning methods on Cityscapes and Pascal VOC 2012. PFB blocks the forward pass of pruned images at the third stage of the encoder. All methods adopt the same training settings.

our method is robust to variations in the number of centroids N_C and feature dimensions D within a normal range.

Moreover, a larger N_C and a lower D help to reduce performance fluctuations. This aligns with intuition: increasing the kernel number within a reasonable range improves KDE estimation accuracy, while a higher dimension increases the complexity and difficulty. We set $N_C=64$ and $D=128$.

5. Conclusion

In this paper, we introduce Partial Forward Blocking (PFB), a novel training data compression paradigm designed to accelerate deep model training. Unlike existing dataset pruning methods that rely on computationally expensive gradient-based importance evaluation or additional proxy models, PFB efficiently selects samples by leveraging a lightweight probability density estimation mechanism without compromising model generalization. Furthermore, we propose a Partial Forward Blocking (PFB) strategy, which skips deep-layer computations for pruned samples, significantly reducing training costs. Extensive experiments

demonstrate that PFB achieves superior training efficiency while outperforming state-of-the-art data pruning and on-line batch selection methods on both classification and segmentation datasets. Our approach provides a scalable and adaptive solution for large-scale dataset training, paving the way for more efficient deep learning models.

6. Acknowledgement

This work was supported by the National Natural Science Foundation of China No.62176097, and the Hubei Provincial Natural Science Foundation of China No.2022CFA055.

References

- [1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 1
- [2] Antoine Bordes, Seyda Ertekin, Jason Weston, and Léon Bottou. Fast kernel classifiers with online and active learning. *Journal of machine learning research*, 6(Sep):1579–1619, 2005. 2
- [3] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022. 1
- [4] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3558–3568, 2021. 1
- [5] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 7
- [6] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018. 6, 12
- [7] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. *arXiv preprint arXiv:1203.3472*, 2012. 2, 6, 7
- [8] Cody Coleman, Christopher Yeh, Stephen Mussmann, Baharan Mirzasoleiman, Peter Bailis, Percy Liang, Jure Leskovec, and Matei Zaharia. Selection via proxy: Efficient data selection for deep learning. In *International Conference on Learning Representations*, 2019. 1, 2, 6, 7
- [9] MMSegmentation Contributors. MMSegmentation: Openmmlab semantic segmentation toolbox and benchmark. <https://github.com/open-mmlab/mms Segmentation>, 2020. 12
- [10] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, pages 3213–3223, 2016. 6, 12
- [11] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018. 12
- [12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6
- [13] Zhijie Deng, Peng Cui, and Jun Zhu. Towards accelerated model training via bayesian data selection. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 3
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6
- [15] Muyang He, Shuo Yang, Tiejun Huang, and Bo Zhao. Large-scale dataset pruning with dynamic uncertainty. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7713–7722, 2024. 2, 6, 7, 12
- [16] Yang He, Lingao Xiao, and Joey Tianyi Zhou. You only condense once: Two rules for pruning condensed datasets. *Advances in Neural Information Processing Systems*, 36: 39382–39394, 2023. 2
- [17] Feng Hong, Yueming Lyu, Jiangchao Yao, Ya Zhang, Ivor Tsang, and Yanfeng Wang. Diversified batch selection for training acceleration. In *Forty-first International Conference on Machine Learning*, 2024. 1, 2, 3, 6, 7, 8, 11, 12
- [18] Angela H Jiang, Daniel L-K Wong, Giulio Zhou, David G Andersen, Jeffrey Dean, Gregory R Ganger, Gauri Joshi, Michael Kaminsky, Michael Kozuch, Zachary C Lipton, et al. Accelerating deep learning by focusing on the biggest losers. *arXiv preprint arXiv:1910.00762*, 2019. 3, 6
- [19] Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR, 2018. 3, 6, 7
- [20] Kenji Kawaguchi and Haihao Lu. Ordered sgd: A new stochastic optimization framework for empirical risk minimization. In *International Conference on Artificial Intelligence and Statistics*, pages 669–679. PMLR, 2020. 6
- [21] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. Glisten: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8110–8118, 2021. 6
- [22] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023. 1
- [23] Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pages 1885–1894. PMLR, 2017. 6
- [24] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 6

- [25] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 10012–10022, 2021. 6
- [26] Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural networks. *arXiv preprint arXiv:1511.06343*, 2015. 1, 3
- [27] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *International Conference on Learning Representations*, 2018. 8
- [28] Sören Mindermann, Jan M Brauner, Muhammed T Razzak, Mrinank Sharma, Andreas Kirsch, Winnie Xu, Benedikt Hölting, Aidan N Gomez, Adrien Morisot, Sebastian Farquhar, et al. Prioritized training on points that are learnable, worth learning, and not yet learnt. In *International Conference on Machine Learning*, pages 15630–15649. PMLR, 2022. 1, 2, 3
- [29] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pages 6950–6960. PMLR, 2020. 6
- [30] Baharan Mirzasoleiman, Kaidi Cao, and Jure Leskovec. Coresets for robust training of deep neural networks against noisy labels. *Advances in Neural Information Processing Systems*, 33:11465–11477, 2020. 2
- [31] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bisaccho, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, page 4. Granada, 2011. 1
- [32] Timothy Nguyen, Zhoung Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression. *International Conference on Learning Representations*, 2020. 1, 2
- [33] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. Deep learning on a data diet: Finding important examples early in training. *Advances in neural information processing systems*, 34:20596–20607, 2021. 1, 2, 6, 7, 8
- [34] Geoff Pleiss, Tianyi Zhang, Ethan Elenberg, and Kilian Q Weinberger. Identifying mislabeled data using the area under the margin ranking. *Advances in Neural Information Processing Systems*, 33:17044–17056, 2020. 6
- [35] Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Daquan Zhou, Lei Shang, Baigui Sun, Xuan-song Xie, Yang You, et al. Infobatch: Lossless training speed up by unbiased dynamic data pruning. In *The Twelfth International Conference on Learning Representations*, 2023. 2, 3, 4, 6, 7, 8, 11, 12
- [36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1
- [37] Ravi S Raju, Kyle Daruwalla, and Mikko Lipasti. Accelerating deep learning with dynamic data pruning. *arXiv preprint arXiv:2111.12621*, 2021. 2, 3, 6, 8
- [38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1
- [39] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015. 1
- [40] David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015. 5, 11
- [41] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017. 2
- [42] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *ICLR*, 2018. 6
- [43] Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018. 5, 11
- [44] Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536, 2022. 1
- [45] Haoru Tan, Sitong Wu, Fei Du, Yukang Chen, Zhibin Wang, Fan Wang, and Xiaojuan Qi. Data pruning via moving-one-sample-out. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 6, 7
- [46] Mariya Toneva, Alessandro Sordani, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2018. 2, 6, 7
- [47] Kai Wang, Bo Zhao, Xiangyu Peng, Zheng Zhu, Shuo Yang, Shuo Wang, Guan Huang, Hakan Bilen, Xinchao Wang, and Yang You. Cafe: Learning to condense dataset by aligning features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12196–12205, 2022. 1, 2
- [48] Xiaobo Xia, Jiale Liu, Jun Yu, Xu Shen, Bo Han, and Tongliang Liu. Moderate coreset: A universal method of data selection for real-world data-efficient deep learning. In *The Eleventh International Conference on Learning Representations*, 2022. 6, 7
- [49] Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li. Dataset pruning: Reducing training data by examining generalization influence. In *The Eleventh International Conference on Learning Representations*, 2022. 1, 2, 6
- [50] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In *10th International Conference on Learning Representations, ICLR 2022*, 2022. 2
- [51] Bo Zhao and Hakan Bilen. Dataset condensation with distribution matching. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6514–6523, 2023. 1

- [52] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. Dataset condensation with gradient matching. In *International Conference on Learning Representations*, 2020. 1
- [53] Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. Coverage-centric coreset selection for high pruning rates. In *11th International Conference on Learning Representations, ICLR 2023*, 2023. 6

A. More Experimental Results

In this section, we provide more experimental results to help our readers better understand our proposed method.

A.1. Ablation Study on the Weight in ADE

To enhance the adaptability of PFB to the changing distribution caused by parameter updating, we introduce the w_j^t (Eq. (4) and Eq. (9)) as a dynamic weight to adaptively balance between different centroids. By re-scaling the output of each kernel according to the number of samples that were previously assigned to its cluster (represented by the centroids), those centroids with a great number of n_j^t will contribute more to the probability density, thereby reducing the importance score of samples with similar features. We evaluate its effectiveness by replacing the w_j^t with 1 (denoted as ‘w/o weight’) and comparing the results with PFB in Tab. 5. Although PFB w/o weight still outperforms DivBS and InfoBatch, the performance drops by a large margin when the prune ratio grows. This phenomenon shows the effectiveness of the balancing weight, which may lead to a better estimation of the probability density function.

Pruning Ratio	30%	50%	70%
Full Data	78.2		
InfoBatch*[35]	78.2 $\uparrow_{0.0}$	78.1 $\downarrow_{0.1}$	76.5 $\downarrow_{1.7}$
DivBS[17]	78.5 $\uparrow_{0.3}$	78.2 $\uparrow_{0.0}$	77.2 $\downarrow_{1.0}$
PFB w/o weight	78.9 $\uparrow_{0.7}$	78.4 $\uparrow_{0.2}$	77.4 $\downarrow_{0.8}$
PFB(ours)	79.1$\uparrow_{0.9}$	78.8$\uparrow_{0.6}$	77.9$\downarrow_{0.3}$

Table 5. Ablation study on the weight in ADE for balancing different centroids. Experiments are conducted on CIFAR-100 using ResNet-18. We use ‘w/o weight’ to denote our modification that w_j^t is replaced by 1 in Eq. (4) of the main text.

A.2. Ablation Study on the Bandwidth Estimation Methods in ADE

As bandwidth estimation is usually deemed important for KDE methods, we compare the performance of two popular bandwidth estimation rules, namely Scott’s rule[40] and Silverman’s rule[43]. We also introduce a baseline denoted as ‘Identity’. This baseline simply sets the \mathbf{H} as an identity matrix. Results in Tab. 6 indicate that a proper bandwidth is crucial for the performance of our PFB. However, there is

Methods	Bandwidth	30%	50%	70%
Full Data	-	78.2		
InfoBatch*[35]	-	78.2 $\uparrow_{0.0}$	78.1 $\downarrow_{0.1}$	76.5 $\downarrow_{1.7}$
DivBS[17]	-	78.5 $\uparrow_{0.3}$	78.2 $\uparrow_{0.0}$	77.2 $\downarrow_{1.0}$
PFB	Identity	77.4 $\downarrow_{0.8}$	76.5 $\downarrow_{1.7}$	75.1 $\downarrow_{3.1}$
	Scott[40]	79.0 $\uparrow_{0.8}$	78.8$\uparrow_{0.6}$	77.9$\downarrow_{0.3}$
	Silverman[43]	79.1$\uparrow_{0.9}$	78.8$\uparrow_{0.6}$	77.9$\downarrow_{0.3}$

Table 6. Ablation study on different bandwidth estimation methods. Experiments are conducted on CIFAR-100 using ResNet-18.

no big difference between those two commonly used bandwidth estimation methods.

A.3. Detailed Explanation of InfoBatch

InfoBatch[35] employs a soft pruning ratio, using the mean loss value of all samples as a threshold to divide the dataset into two subsets. Samples with a loss lower than this threshold form a candidate subset, where each sample has a pruning probability of p , which is reported as the pruning ratio. However, since the candidate subset only contains half of the samples, the actual pruning ratio of InfoBatch is $p/2$. To highlight this distinction, we denote the original version of InfoBatch with ‘*’ in the main text. For a fair comparison, we follow the modification applied by DivBS[17] to InfoBatch, where the threshold is set to the 95% percentile to align with the actual pruning ratio of most pruning methods. We denote this modified version as InfoBatch † . In the main text, Tab. 2-4 present a comparison between our method and InfoBatch † on ImageNet-1k, Cityscapes, and PASCAL VOC 2012. Here, we further provide experimental results on CIFAR-100 in Tab. 7 to supplement the comparison. Aligning the actual pruning ratio reveals a significant performance drop for InfoBatch on CIFAR-100. Beyond the impact of the adjusted pruning ratio, this decline may also stem from the large weights applied to the retained samples within the pruned candidate subset by InfoBatch. (Please refer to [35] for details of this re-scaling operation.) Such a large weight may excessively emphasize the retained samples, potentially hindering the learning of harder examples in the other subset.

A.4. Error Bars

Error statistics for ResNet-18 and Swin-T on different datasets are also included in Tab. 8, exhibiting variations within an acceptable range. The results show that PFB can maintain a stable performance with negligible variation.

B. Implementation Details

We further demonstrate the details of experiments on image classification and segmentation datasets here.

Pruning Ratio	30%	50%	70%
Actual Ratio	15%	25%	35%
InfoBatch*[35]	78.2 $\uparrow_{0.0}$	78.1 $\downarrow_{0.1}$	76.5 $\downarrow_{1.7}$
Pruning Ratio	30%	50%	70%
InfoBatch †	78.0 $\downarrow_{0.2}$	76.0 $\downarrow_{2.2}$	74.3 $\downarrow_{3.9}$
DivBS[17]	78.5 $\uparrow_{0.3}$	78.2 $\uparrow_{0.0}$	77.2 $\downarrow_{1.0}$
PFB(ours)	79.1$\uparrow_{0.9}$	78.8$\uparrow_{0.6}$	77.9$\downarrow_{0.3}$
Full Data	78.2		

Table 7. Fair comparison on CIFAR-100 with InfoBatch.

Dataset/Model	CIFAR-10 / ResNet-18			CIFAR-100 / ResNet-18			ImageNet-1K / Swin-T		
Pruning Ratio	30%	50%	70%	30%	50%	70%	30%	40%	50%
PFB(Ours)	95.9	95.5	95.2	79.1	78.8	77.9	79.6	79.2	78.2
	± 0.1	± 0.1	± 0.2	± 0.2	± 0.2	± 0.2	± 0.1	± 0.2	± 0.2
Full Data	95.6 ± 0.1			78.2 ± 0.1			79.6 ± 0.1		

Table 8. Error bars on CIFAR-10/100 and ImageNet-1k.

B.1. Classification Training Settings

All the classification experiments are conducted on a 4-RTX 4090 GPU server. We follow the training details of InfoBatch[35] on CIFAR-10/100 using ResNet-18 and ImageNet-1k using ResNet-50. For Swin-T, we adopt similar training settings of Dyn-Unc[15]. The AutoAugment[11] is applied to augment training data only for Swin-T, including random path drop and gradient clipping for a fair comparison with Dyn-Unc[15] in Tab. (2) of the main text. All the detailed settings needed for reproduction are listed in the Tab. 9.

B.2. Details of Segmentation Experiments

Our segmentation experiments are based on the implementation of MMSegmentation[9]. On PASCAL VOC 2012[6] and Cityscapes[10], the models are trained for 36,000 iterations. Other training and evaluation details remain the same with MMSegmentation. Please note that the reported mIoU results in Tab. (3) and (4) employ the popular multi-scale evaluation technique. Moreover, as most semantic segmentation methods employ an auxiliary segmentation head at the third stage of the encoder, we extract the features at this stage to utilize the abundant semantic information. Please note that there is a big difference between segmentation and classification networks: aside from the encoder, segmentation networks usually have a computationally expensive decoder. Hence, blocking at the third stage of the encoder can still significantly cut down the training time.

Parameters	CIFAR-10	CIFAR-100	ImageNet-1k	
Models	ResNet-18	ResNet-18	ResNet-50	Swin-T
Training	optimizer	SGD	SGD	Lars
	weight_decay	0.0005	0.0005	AdamW
	batch_size	128	128	1024
	epochs	200	200	90
	learning_rate	0.10	0.05	6.4
	label smoothing	0.1	0.1	0.1
	learning rate scheduler	OncCycle	OncCycle	OncCycle
	learning rate warmup	-	-	5
Data Pruning	b	0.01	0.01	0.001
	N_C	64	64	64
	D	128	128	128
	PFB Location	stage-1	stage-1	stage-2
	epoch start pruning	5	5	15
	epoch stop pruning	180	180	80

Table 9. Detailed training settings on image classification datasets.