Randomly Pivoted V-optimal Design: Fast Data Selection under Low Intrinsic Dimension

Yijun Dong*
Courant Institute
New York University
yd1319@nyu.edu

Xiang Pan*
Center of Data Science
New York University
xiangpan@nyu.edu

Hoang Phan*
Center of Data Science
New York University
hvp2011@nyu.edu

Qi Lei
Center of Data Science
New York University
ql518@nyu.edu

Abstract

Despite the ubiquitous high-dimensionalities brought about by the increasing sizes of models and data, low intrinsic dimensions are commonly found in many high-dimensional learning problems (*e.g.* finetuning). To explore sample efficient learning that leverages such low intrinsic dimensions, we introduce randomly pivoted V-optimal design (RPVopt), a fast data selection algorithm that combines dimension reduction via sketching and optimal experimental design. Given a large dataset with N samples in a high dimension d, RPVopt first reduces the dimensionality from d to $m \ll d$ by embedding the data to a random low-dimensional subspace via sketching. Then a coreset of size $n > m$ is selected based on the low-dimensional sketched data through an efficient two-stage random pivoting algorithm. With a fast embedding matrix for sketching, RPVopt achieves an asymptotic complexity of $O(Nd + Nnm)$, linear in the full data size, data dimension, and coreset size. With extensive experiments in both regression and classification settings, we demonstrate the empirical effectiveness of RPVopt in data selection for finetuning vision tasks.

1 Introduction

Deep learning models have achieved remarkable success across various domains, including vision [1, 2] and languages [3, 4]. These large models typically require training on astronomical-scale datasets [5, 6]. However, the computational costs and data storage demands associated with such datasets pose substantial challenges. Consequently, there is increasing interest in enhancing data efficiency and reducing dataset sizes without sacrificing performance. Prominent strategies such as coreset selection and data condensation have emerged. By identifying and retaining a condensed yet representative sample set from larger datasets, these techniques allow the training of models on smaller, yet representative, datasets that aim to mirror the learning potential of the full dataset.

Despite the increasing dimensionalities in modern machine learning, low intrinsic dimensions can often be found in many high-dimensional learning problems like finetuning [7, 8]. Inspired by this seminal work, there are follow-up works [9, 10] employing a low-dimensional reparameterization for parameter-efficient finetuning. Such low intrinsic dimensions suggest that, under suitable regularization, learning with a small subset of the huge original dataset should be sufficient to mimic the performance of full-size training. Recent work also shows that compressing the model via intrinsic

*Alphabetical order.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

dimension yields better results than standard pruning and uses them to derive compression-based generalization bounds [11].

For statistical models, data selection is often formulated as an optimal experimental design (OED) problem [12, 13, 14, 15]. In the classical overdetermined setting for OED (where the problem dimension d is lower than the data size n), V(ariance)-optimality is a design criterion tailored to control the generalization gap. Inspired by the recent progress [16] on extending the notion of V-optimality to overparametrized data selection with low intrinsic dimension via sketching, we introduce Randomly Pivoted V-optimal Design (RPVopt), a fast data selection method for learning under low intrinsic dimension.

Concretely, RPVopt first explores the low intrinsic dimension by embedding the high-dimensional problem to a random low-dimensional subspace via sketching [16]. After reducing to the classical low-dimensional (overdetermined) setting, in contrast to the common practice of solving an expensive continuous relaxation of the discrete optimization problem posed by V-optimality [17, 18, 16], we introduce a more efficient two-stage random pivoting algorithm that samples the coreset adaptively. For the full data size N, problem dimension d, coreset size n, and embedding dimension m, RPVopt runs in $O(Nd + Nnm)$ time with a fast embedding for sketching. Via extensive experiments on fine-tuning vision models, we empirically validate the performance of RPVopt in both regression and classification settings, where it outperforms existing data selection methods across various settings, especially for small coresets.

1.1 Related Works

Due to the space limit, we focus on OED here and defer further discussions regarding the more general data selection problem to Appendix A.1.

Optimal experimental design. While standard OED focuses on the overdetermined scenario with $n \geq d$, efforts have been made to extend the notion of V-optimality to overparametrized setting, $d > n$ [19, 20, 16]. Specifically, [19, 20] proposes design criteria for ridge regression in the general overparametrized setting. More recently, [16] considers overparametrized ridge regression with low intrinsic dimension in the context of finetuning and provides a selection criterion based on sketching that brings a sample complexity independent of d.

Fast algorithms for V-optimality. Despite the long history of OED, progress in provable algorithms for V-optimality [18, 17, 21] has taken place more recently and remains surprisingly sparse. In particular, [17, 21] introduced an optimization-based framework for a broad variety of optimality criteria, including the V-optimality, which provably finds a nearly optimal coreset in polynomial time. The framework consists of two stages: (i) solving a continuous relaxation of the original discrete optimization problem and (ii) rounding the continuous solution via regret minimization. However, solving the continuous relaxation can be prohibitively expensive despite its polynomial complexity [21]. Related to V-optimality, A(verage)-optimality is a more studied design criterion. Specifically, [22] shows that under mild conditioning assumptions, the classical Fedorov's exchange method [14] finds a nearly optimal coreset in polynomial time. Beyond computational tractability, [23] investigates and improves a set of fast algorithms for the A-optimal experimental design, including greedy removal [24], volume sampling [24], leverage score sampling [25], and dual set sparsification [26].

2 Data Selection under Low Intrinsic Dimension

Notations. Given any $n \in \mathbb{Z}_+$, let $[n] = \{1, \cdots, n\}$. Let $\mathbf{e}_n$ be the n-th canonical basis of the conformable dimension.For any set U, we denote $|U|$ as the cardinality of U. Additionally, for any $n \in [|U|]$, let $\mathcal{C}(U, n) = \{S \subseteq U \mid |S| = n\}$. We adapt the standard asymptotic notations: for any functions $f, g : \mathbb{R}_+ \to \mathbb{R}_+$, we write $f = O(g)$ or $f \lesssim g$ if there exists some constant $C > 0$ such that $f(x) \leq Cg(x)$ for all $x \in \mathbb{R}_+$; $f = \Omega(g)$ or $f \gtrsim g$ if $g = O(f)$; $f \asymp g$ if $f = O(g)$ and $f = \Omega(g)$. For any matrix $\mathbf{A}$, let $\sigma_1(\mathbf{A}) \geq \cdots \geq \sigma_{\mathrm{rank}(\mathbf{A})}(\mathbf{A}) \geq 0$ be the singular values; and denote $\mathbf{A}^\dagger$ as the Moore-Penrose pseudoinverse. Additionally for any $k \leq \mathrm{rank}(\mathbf{A})$, let $\langle \mathbf{A} \rangle_k = \mathrm{argmin}_{\mathbf{B}:\, \mathrm{rank}(\mathbf{B}) \leq k} \|\mathbf{A} - \mathbf{B}\|_F$ be the optimal rank-k approximation of $\mathbf{A}$ (characterized by the rank-k truncated SVD). For any symmetric matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{d \times d}$, we write $\mathbf{A} \succcurlyeq \mathbf{B}$ or $\mathbf{A} - \mathbf{B} \succcurlyeq 0$ if $\mathbf{A} - \mathbf{B}$ is positive semidefinite.

2.1 Low-dimensional Data Selection and V-optimal Design

Data distribution. Consider a data distribution P over $\mathcal{X}\times\mathbb{R}$ ($\mathcal{X}\subset\mathbb{R}^d$) characterized by the ground truth $\boldsymbol{\theta}_*\in\mathbb{R}^d$ and level of noise $\sigma>0$: (i) $\mathbb{E}_{(\mathbf{x},y)\sim P}[y\mid\mathbf{x}]=\mathbf{x}^\top\boldsymbol{\theta}_*$, and (ii) $\mathbb{V}_{(\mathbf{x},y)\sim P}[y\mid\mathbf{x}]\le\sigma^2$. Let $\mathbf{X}=[\mathbf{x}_1,\cdots,\mathbf{x}_N]^\top\in\mathbb{R}^{N\times d}$ be the data matrix associated with a huge set of N unlabeled samples $\left\{\mathbf{x}_i\in\mathbb{R}^d\;\middle|\;i\in[N]\right\}$ drawn *i.i.d.* from P. For simplicity, we assume $\operatorname{rank}(\mathbf{X})=d$ and consider the fixed design setting with $\mathcal{X}=\mathbf{X}$ and a uniform marginal distribution $P(\mathbf{x}_i)=1/N$ for all $i\in[N]$. Each $\mathbf{x}_i$ is associated with an unknown label $y_i=\mathbf{x}_i^\top\boldsymbol{\theta}_*+z_i$ that can be queried with non-negligible cost, where z_i is an independent and zero-mean random variable with $\mathbb{V}[z_i]\le\sigma^2$.

Learning problem. For any $S=\{i_1,\cdots,i_n\}\in\mathcal{C}([N],n)=\{S\subset[N]\mid|S|=n\}$, let $\mathbf{X}_S=[\mathbf{x}_{i_1},\cdots,\mathbf{x}_{i_n}]^\top\in\mathbb{R}^{n\times d}$ be the data submatrix selected by S. Denote $\boldsymbol{\Sigma}=\frac{1}{N}\mathbf{X}^\top\mathbf{X}$ and $\boldsymbol{\Sigma}_S=\frac{1}{n}\mathbf{X}_S^\top\mathbf{X}_S$ as the $d\times d$ second moments. Given a target coreset size $n<N$, the goal of data selection is to find a nearly optimal data subset indexed by $S\in\mathcal{C}([N],n)$ such that by querying only n labels $\mathbf{y}_S\in\mathbb{R}^n$ associated with $\mathbf{X}_S$, one can learn a "good" approximation $\boldsymbol{\theta}_S$ of $\boldsymbol{\theta}_*$ from $(\mathbf{X}_S,\mathbf{y}_S)$. Consider a regression problem with ℓ_2 population loss, $L(\boldsymbol{\theta})=\mathbb{E}_{(\mathbf{x},y)\sim P}[(\mathbf{x}^\top\boldsymbol{\theta}-y)^2]$. In the context of statistical learning, a "good" approximation of $\boldsymbol{\theta}_*$ generally refers to a $\boldsymbol{\theta}\in\mathbb{R}^d$ with low excess risk, $\mathrm{ER}(\boldsymbol{\theta})=L(\boldsymbol{\theta})-L(\boldsymbol{\theta}_*)=\|\boldsymbol{\theta}-\boldsymbol{\theta}_*\|_{\boldsymbol{\Sigma}}^2$, where $\boldsymbol{\Sigma}\succ0$ as $\operatorname{rank}(\mathbf{X})=d$.

Low- v.s. high-dimensional data selection. We refer to *"low-dimensional" data selection* as the case where the data dimension is lower than the coreset size, $d\le n$ (or more precisely, $d=\operatorname{rank}(\mathbf{X}_S)$), and therefore, $\boldsymbol{\theta}_S$ is uniquely identified by an overdetermined system:

$$\text{Low-dimensional}:\ d=\operatorname{rank}(\mathbf{X}_S),\quad \boldsymbol{\theta}_S=\operatorname*{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^d}\frac{1}{n}\|\mathbf{X}_S\boldsymbol{\theta}-\mathbf{y}_S\|_2^2 \tag{1}$$

In contrast, *"high-dimensional" data selection* refer to an overparametrized problem with $d>n$ (or more precisely, $d>\operatorname{rank}(\mathbf{X}_S)$), where $\boldsymbol{\theta}_S$ is learned through ridge regression with a suitable choice of regularization hyperparameter $\alpha>0$:

$$\text{High-dimensional}:\ d>\operatorname{rank}(\mathbf{X}_S),\quad \boldsymbol{\theta}_S=\operatorname*{argmin}_{\boldsymbol{\theta}\in\mathbb{R}^d}\frac{1}{n}\|\mathbf{X}_S\boldsymbol{\theta}-\mathbf{y}_S\|_2^2+\alpha\|\boldsymbol{\theta}\|_2^2. \tag{2}$$

V-optimal design. Classical OED studies the low-dimensional data selection problem where various optimality criteria [13] are introduced to characterize different notions of "distance" between $\boldsymbol{\theta}_S$ and $\boldsymbol{\theta}_*$. For example, the A(verage)-optimality $\operatorname{tr}(\boldsymbol{\Sigma}_S^\dagger)$ is associated with the Euclidean distance $\|\boldsymbol{\theta}_S-\boldsymbol{\theta}_*\|_2^2$; whereas the V(ariance)-optimality $\operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{\Sigma}_S^\dagger)$ is arguably the most relevant criterion that directly controls the excess risk (*e.g.* see [15, Section 7.5.2] or [16, (1)]):

$$\mathbb{E}\left[\mathrm{ER}(\boldsymbol{\theta}_S)\right]=\mathbb{E}\left[\|\boldsymbol{\theta}_S-\boldsymbol{\theta}_*\|_{\boldsymbol{\Sigma}}^2\right]\le\frac{\sigma^2}{n}\operatorname{tr}(\boldsymbol{\Sigma}\boldsymbol{\Sigma}_S^\dagger). \tag{3}$$

2.2 High-dimensional Data Selection under Low Intrinsic Dimension

For data selection, a common and intriguing high-dimensional setting is learning problems with low intrinsic dimensions [7, 8] (*e.g.* finetuning). Specifically, [8] unveils the possibility of finetuning high-dimensional models with sample complexities proportional to their low intrinsic dimensions, which is impossible in general high-dimensional settings.

For high-dimensional data selection under low intrinsic dimension, [16] proposes to (i) find a low-dimensional subspace that encapsulates crucial information in data via sketching [27, 28, 29], and then (ii) select data by solving the OED problem in the resulting low-dimensional subspace. In this context, [16] introduces a data selection criterion that generalizes the notion of V-optimality:

Remark 2.1 ([16, Theorem 3.1])**.** *Let $\overline{r}=\min\{t\in[r]\mid\|\langle\mathbf{X}\rangle_t\|_F^2\ge(1-\frac{1}{N})\|\mathbf{X}\|_F^2\}$ be the intrinsic dimension of the dataset $\mathbf{X}$. Assume $\mathbf{X}$ has a low intrinsic dimension: $\overline{r}\ll\min\{N,d\}$. Sketch $\mathbf{X}$ via a Gaussian embedding $\boldsymbol{\Gamma}\in\mathbb{R}^{d\times m}$ with* i.i.d. *entries $\boldsymbol{\Gamma}_{ij}\sim\mathcal{N}(0,1/m)$ and embedding dimension $m\ge11\overline{r}$ such that $\widetilde{\boldsymbol{\Sigma}}=\frac{1}{N}(\mathbf{X}\boldsymbol{\Gamma})^\top(\mathbf{X}\boldsymbol{\Gamma})$ and $\widetilde{\boldsymbol{\Sigma}}_S=\frac{1}{n}(\mathbf{X}_S\boldsymbol{\Gamma})^\top(\mathbf{X}_S\boldsymbol{\Gamma})$ for any $S\in\mathcal{C}([N],n)$. If $\sigma_{\lceil1.1\overline{r}\rceil}(\widetilde{\boldsymbol{\Sigma}}_S)\ge\gamma_S$ for some $\gamma_S>0$, then with probability at least 0.9 over $\boldsymbol{\Gamma}$, there exists a regularization hyperparameter $\alpha>0$ such that* (2) *satisfies*

$$\mathbb{E}\left[\mathrm{ER}\left(\boldsymbol{\theta}_S\right)\right]\lesssim\underbrace{\frac{\sigma^2}{n}\operatorname{tr}(\widetilde{\boldsymbol{\Sigma}}(\widetilde{\boldsymbol{\Sigma}}_S)^\dagger)}_{\textit{variance}}+\underbrace{\frac{\sigma^2}{n}\frac{1}{m\gamma_S}\|\widetilde{\boldsymbol{\Sigma}}(\widetilde{\boldsymbol{\Sigma}}_S)^\dagger\|_2\operatorname{tr}(\boldsymbol{\Sigma})}_{\textit{sketching error}}+\underbrace{\frac{1}{n}\|\widetilde{\boldsymbol{\Sigma}}(\widetilde{\boldsymbol{\Sigma}}_S)^\dagger\|_2\operatorname{tr}(\boldsymbol{\Sigma})\|\boldsymbol{\theta}_*\|_2^2}_{\textit{bias}}. \tag{4}$$

In particular, when $\|\widetilde{\mathbf{\Sigma}}(\widetilde{\mathbf{\Sigma}}_S)^\dagger\|_2 \le c_S$, under mild regularity assumptions $\sigma^2 = O(1)$, $\|\boldsymbol{\theta}_*\|_2^2 = O(1)$, and $\mathrm{tr}(\mathbf{\Sigma}) = O(\overline{r})$, taking $m = \max\{\sqrt{\mathrm{tr}(\mathbf{\Sigma})/\gamma_S}, 11\overline{r}\}$ leads to $\mathbb{E}\left[\mathrm{ER}\left(\boldsymbol{\theta}_S\right)\right] \lesssim c_S\overline{r}/n$, *i.e.* a sample complexity proportional to the low intrinsic dimension $\overline{r}$.

3 Randomly Pivoted V-optimal Design

Observing that the generalization of data selection in (4) is governed by $\mathrm{tr}(\widetilde{\mathbf{\Sigma}}(\widetilde{\mathbf{\Sigma}}_S)^\dagger) \ge \|\widetilde{\mathbf{\Sigma}}(\widetilde{\mathbf{\Sigma}}_S)^\dagger\|_2$, in this section, we introduce a fast and effective data selection algorithm based on sketching and random pivoting that adaptively samples data to optimize $\mathrm{tr}(\widetilde{\mathbf{\Sigma}}(\widetilde{\mathbf{\Sigma}}_S)^\dagger)$ locally.

Dimension reduction via sketching. Algorithm 3.1 starts by embedding the high-dimensional data to a random low-dimensional subspace via sketching: $\widetilde{\mathbf{X}} = \mathbf{X}\mathbf{\Gamma} \in \mathbb{R}^{N\times m}$, where a common choice of $\mathbf{\Gamma}$ is a Gaussian random matrix (vide [27, 28] for a comprehensive overview of sketching).

Randomly pivoted QR. With $\widetilde{\mathbf{X}}$, Algorithm 3.1 selects the first m samples via randomly pivoted QR [30]: Initializing $\widetilde{\mathbf{X}}^{(0)} = [\widetilde{\mathbf{x}}_1^{(0)}, \cdots, \widetilde{\mathbf{x}}_N^{(0)}]^\top = \widetilde{\mathbf{X}} \in \mathbb{R}^{N\times m}$ and $S_0 = \emptyset$, for $t \in [m]$:

(i) Sample i_t from $[N] \setminus S_{t-1}$ with probability $p_i = \|\widetilde{\mathbf{x}}_i^{(t-1)}\|_2^2/\|\widetilde{\mathbf{X}}^{(t-1)}\|_F^2$ for all $i \in [N] \setminus S_{t-1}$;

(ii) Update $S_t = S_{t-1} \cup \{i_t\}$ and $\widetilde{\mathbf{X}}^{(t)} = \widetilde{\mathbf{X}}^{(t-1)} - \widetilde{\mathbf{X}}^{(t-1)}\widetilde{\mathbf{x}}_{i_t}^{(t-1)}(\widetilde{\mathbf{x}}_{i_t}^{(t-1)})^\top/\|\widetilde{\mathbf{x}}_{i_t}^{(t-1)}\|_2^2$.[2]

When $\mathrm{rank}(\mathbf{X}) = d$, sketching via a Gaussian embedding with $m \le d$ provides $\mathrm{rank}(\widetilde{\mathbf{X}}) = m$ with probability one. Then, the Gram-Schmidt process ensures that randomly pivoted QR selects m linearly independent samples, *i.e.* $\mathrm{rank}(\widetilde{\mathbf{X}}_{S_m}) = m$. It is worth noticing that random pivoted QR is effectively replacing the greedy pivoting in the classical row pivoted QR [32] with adaptive square norm sampling, which achieves better accuracy and robustness in both theory and practice [30].

Adaptive V-optimal design via random pivoting. With the first m linearly independent samples, Algorithm 3.1 continues by adaptively sampling the remaining $n - m$ data according to the V-optimality over $\widetilde{\mathbf{X}}$. In particular, since $\mathrm{rank}(\widetilde{\mathbf{X}}_{S_m}) = m$, for any subsequent $S \supset S_m$, $\mathrm{rank}(\widetilde{\mathbf{X}}_S) = m$. Then, the Woodbury matrix identity [33] implies that for any $S \supset S_m$ and $i \in [N] \setminus S$,

$$(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S + \widetilde{\mathbf{x}}_i\widetilde{\mathbf{x}}_i^\top)^{-1} = (\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1} - \frac{(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1}\widetilde{\mathbf{x}}_i\widetilde{\mathbf{x}}_i^\top(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1}}{1 + \widetilde{\mathbf{x}}_i^\top(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1}\widetilde{\mathbf{x}}_i}, \tag{5}$$

and therefore, $\mathrm{tr}(\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}}(\widetilde{\mathbf{X}}_{S\cup\{i\}}^\top\widetilde{\mathbf{X}}_{S\cup\{i\}})^{-1}) = \mathrm{tr}(\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}}(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1}) - \Delta_S(\widetilde{\mathbf{x}}_i)$ where $\Delta_S(\widetilde{\mathbf{x}}_i) = \|\widetilde{\mathbf{X}}(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1}\widetilde{\mathbf{x}}_i\|_2^2/(1 + \widetilde{\mathbf{x}}_i^\top(\widetilde{\mathbf{X}}_S^\top\widetilde{\mathbf{X}}_S)^{-1}\widetilde{\mathbf{x}}_i)$. Since $\mathrm{tr}(\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{\Sigma}}_{S_n}^{-1}) = \frac{n}{N}\mathrm{tr}(\widetilde{\mathbf{X}}^\top\widetilde{\mathbf{X}}(\widetilde{\mathbf{X}}_{S_n}^\top\widetilde{\mathbf{X}}_{S_n})^{-1})$, for given $S \supset S_m$, picking $i \in [N] \setminus S$ with the maximum $\Delta_S(\widetilde{\mathbf{x}}_i)$ brings the optimal reduction in $\mathrm{tr}(\widetilde{\mathbf{\Sigma}}\widetilde{\mathbf{\Sigma}}_{S_n}^{-1})$ locally, leading to a natural greedy algorithm.

To alleviate the potential suboptimality caused by the local optimality, instead of greedy selection, we inject randomness by sampling proportional to $\exp(\Delta_{S_{t-1}}(\widetilde{\mathbf{x}}_i)/\tau)$, with the randomness controlled by a temperature hyperparameter τ.

Notice that the overall asymptotic complexity of Algorithm 3.1 is $O(Ndm + Nnm)$ with the Gaussian embedding, whereas leveraging more efficient sparse embeddings [34, 35, 36] can further bring down the complexity to $O(\mathrm{nnz}(\mathbf{X}) + Nnm)$ in practice [37], where $\mathrm{nnz}(\mathbf{X}) \le Nd$ denotes the number of nonzero entries in $\mathbf{X}$. This matches the complexity of ridge leverage score sampling with fast leverage score approximation [38, 39, 40], which (to the best of our knowledge) is one of the most efficient provable algorithms for A-optimality [23] and data selection [39]. In Section 4.1, we empirically demonstrate that RPVopt outperforms ridge leverage score sampling in data selection for regression, especially when the coreset size is small. The theoretical guarantee for RPVopt that matches its empirical performance remains an exciting open problem.

4 Experiments

In this section, we evaluate the performance of RPVopt in Algorithm 3.1 on different settings. We first show the effectiveness of our proposed method on regression tasks then extend the experimental

[2]This Gram-Schmidt process is numerically unstable with floating point arithmetic. In practice, a stable implementation like [31, Algorithm 3.1] is used.

Algorithm 3.1 Randomly Pivoted V-optimal Design (RPVopt)

Input: $\mathbf{X} \in \mathbb{R}^{N\times d}$, coreset size n, temperature $\tau > 0$, embedding dimension $m < n$.

(Draw a Gaussian embedding $\mathbf{\Gamma} \in \mathbb{R}^{d\times m}$ with *i.i.d.* entries $\mathbf{\Gamma}_{ij} \sim (0, 1/m)$.)

(Compute the sketching $\widetilde{\mathbf{X}} = \mathbf{X}\mathbf{\Gamma} = [\widetilde{\mathbf{x}}_1, \cdots, \widetilde{\mathbf{x}}_N]^\top \in \mathbb{R}^{N\times m}$.) ▷ $O(Ndm)$

Select $S_m \in \mathcal{C}([N], m)$ from $\widetilde{\mathbf{X}}$ via randomly pivoted QR. ▷ $O(Nm^2)$

$\mathbf{Y}_{(m)} \leftarrow \widetilde{\mathbf{X}}(\widetilde{\mathbf{X}}_{S_m}^\top \widetilde{\mathbf{X}}_{S_m})^{-1} \in \mathbb{R}^{N\times m}$. ▷ $O(Nm^2 + m^3)$

for $t = m+1, \cdots, n$ **do**

  $\Delta_{S_{t-1}}(\widetilde{\mathbf{x}}_i) \leftarrow \|\mathbf{Y}_{(t-1)}\widetilde{\mathbf{x}}_i\|_2^2/(1 + \mathbf{e}_i^\top \mathbf{Y}_{(t-1)}\widetilde{\mathbf{x}}_i)\ \forall\ i \in \overline{S_{t-1}}$. ▷ $O(Nm)$

  $\mathbf{p} = (p_1, \cdots, p_N) \leftarrow \mathbf{0}_{[N]}$, $p_i \leftarrow \exp(\Delta_{S_{t-1}}(\widetilde{\mathbf{x}}_i)/\tau)\ \forall\ i \in \overline{S_{t-1}}$.

  Sample $i_t \sim \mathbf{p}/\|\mathbf{p}\|_1$

  $S_t \leftarrow S_{t-1} \cup \{i_t\}$.

  $\mathbf{Y}_{(t)} \leftarrow \mathbf{Y}_{(t-1)} - (\mathbf{Y}_{(t-1)}\widetilde{\mathbf{x}}_{i_t})(\mathbf{e}_{i_t}^\top \mathbf{Y}_{(t-1)})/(1 + \mathbf{e}_{i_t}^\top \mathbf{Y}_{(t-1)}\widetilde{\mathbf{x}}_{i_t})$. ▷ $O(Nm)$

return $S_n \in \mathcal{C}([N], n)$.

protocol to include classification tasks. Due to the limit constraint, the detailed configurations are provided in Appendix B.

4.1 Regression

Datasets and network architectures. We conduct regression experiments on UTKFace [41], which is a dataset for age estimation, with CLIP [1] and ResNet18 [42] backbones. While we examine linear probing on CLIP (ViT-B/32), we fine-tune the projections layer and the classifier of ResNet18 to represent the low- and high-dimensional settings, respectively. For both experiments, we utilize the Adam optimizer [43] with a batch size of 128 and an initial learning rate of 0.1.

Baselines. We evaluate our method by comparing it against notable unsupervised data selection methods for regression: (1) **Uniform Sampling** randomly all samples with equal probability, (2) **Adaptive Sampling** [44, 30] progressively sampling data based on their squared norms and adaptively eliminating the spanning subspace of the selected samples, (3) **Ridge Leverage Score Sampling** [38, 39, 40] extending classical leverage score sampling [45] to high dimensions, measuring of the influence that individual data points have on linear regression models, (4) **Greedy** [46] choosing a subset such that the bound between an average loss over any given subset of the dataset and the remaining data points is minimized, (5) **Herding** [47] greedily selects samples to minimize the selection set center and full dataset center in the feature space.

Table 1: Mean Absolute Error (the lower the better) on UTKFace with a linear regressor trained on top of frozen features from a pre-trained CLIP (ViT-B/32). We use the **bold** font to indicate the best method for each coreset size.

Method	100	200	500	1000	2000	3000
Uniform Sampling	10.55 ± 3.09	8.94 ± 3.48	6.09 ± 0.42	4.70 ± 0.23	3.92 ± 0.16	3.68 ± 0.15
Adaptive	6.02 ± 0.53	4.75 ± 0.14	4.40 ± 0.14	N/A	N/A	N/A
Greedy	10.40 ± 1.21	7.56 ± 0.18	6.43 ± 0.09	5.51 ± 0.19	4.87 ± 0.03	4.37 ± 0.08
Herding	17.57 ± 0.01	13.41 ± 0.01	8.47 ± 0.01	5.79 ± 0.01	4.19 ± 0.01	3.53 ± 0.01
R-leverage	5.44 ± 0.01	4.79 ± 0.02	4.36 ± 0.01	3.86 ± 0.01	$\mathbf{3.61 \pm 0.01}$	3.53 ± 0.04
RPVopt	$\mathbf{5.14 \pm 0.30}$	$\mathbf{4.43 \pm 0.12}$	$\mathbf{4.13 \pm 0.24}$	$\mathbf{3.82 \pm 0.07}$	3.67 ± 0.05	$\mathbf{3.47 \pm 0.14}$

The results for linear probing are provided in Table 1, where our method remarkably outperforms comparative baselines on UTKFace, especially for smaller coreset sizes n. For $n = 100, 200, 500$, RPVopt exceeds the performance of the second-best method, R-leverage, by approximately 0.3 MAE, and achieves a Mean Absolute Error reduction of $30 - 50\%$ compared to Uniform Sampling.

Table 2: Mean Absolute Error on UTK in fine-tuning the last two layers of ResNet18.

Method	100	200	500	1000	2000	3000
Uniform Sampling	8.43 ± 1.54	8.13 ± 0.52	6.62 ± 0.38	5.44 ± 0.53	5.02 ± 0.67	4.40 ± 0.49
Adaptive	9.60 ± 0.10	8.29 ± 0.91	6.28 ± 0.77	N/A	N/A	N/A
Greedy	10.82 ± 1.29	9.83 ± 0.51	6.98 ± 0.71	5.95 ± 0.22	5.33 ± 0.74	4.60 ± 0.19
Herding	23.08 ± 2.11	22.33 ± 1.49	19.68 ± 0.02	20.24 ± 0.13	6.31 ± 1.49	5.39 ± 0.58
R-leverage	12.20 ± 0.19	10.68 ± 0.37	7.71 ± 0.33	5.50 ± 0.44	5.25 ± 0.42	$\mathbf{4.20 \pm 0.44}$
RPVopt	$\mathbf{8.39 \pm 0.19}$	$\mathbf{7.33 \pm 0.61}$	$\mathbf{5.74 \pm 0.31}$	$\mathbf{4.74 \pm 0.16}$	$\mathbf{4.58 \pm 0.33}$	4.32 ± 0.27

In Table 2, we finetune the last two layers of ResNet18 [42] using the same optimization setting with the above linear probing experiment. Similarly, RPVopt consistently achieves the best performance

among all baselines. It is worth noting that while some selected comparative underperform relative to the uniform sampling baseline, particularly at higher pruning rates, our method consistently surpasses this baseline across various coreset sizes. For the UTKFace experiments, the computational cost associated with the Adaptive method is prohibitively expensive, rendering it impractical for large core sizes, i.e., greater than 500.

4.2 Classification

To evaluate the performance of our methods beyond the regression task, we conduct experiments on the CIFAR10 [48] and StanfordCars [49] datasets in the classification setting. Apart from those baselines described in the above experiments, we compare our methods with the following methods based on the DeepCore benchmark [50]. **Contextual Diversity** [51] proposes using contextual diversity to select diverse samples. **Glister** [52] uses bi-level optimization to optimize the selection set. **GraNd** [53] uses the gradient norm of warmup trained model to select samples. **Forgetting** [54] uses the forgetting events (correctly classified samples that are later misclassified during the training process) as the selection criterion. **DeepFool** [55] uses the adversarial attacking strength to identify the samples that are close to the decision boundary and select them. **Uncertainty-Based Methods** [56] use the model's uncertainty metric (entropy, margin, confidence) to construct the selection set.

Table 3: Linear Probing performance of CLIP on StanfordCars with different data pruning methods.

Method	Metric	500	1000	1500	2000	2500	3000	3500	4000
Uniform Sampling	Acc	38.90 ± 0.46	54.60 ± 0.46	62.60 ± 0.23	67.63 ± 0.17	70.59 ± 0.19	72.49 ± 0.19	74.16 ± 0.22	75.40 ± 0.16
	F1	32.30 ± 0.43	49.94 ± 0.56	58.99 ± 0.23	64.54 ± 0.18	67.79 ± 0.23	70.00 ± 0.20	71.77 ± 0.23	73.14 ± 0.12
Herding [47]	Acc	38.86 ± 0.40	54.95 ± 0.33	63.44 ± 0.31	67.22 ± 0.16	71.02 ± 0.13	73.17 ± 0.22	74.64 ± 0.18	75.71 ± 0.29
	F1	31.80 ± 0.32	50.14 ± 0.51	59.75 ± 0.32	64.07 ± 0.23	68.28 ± 0.15	70.64 ± 0.28	72.22 ± 0.26	73.26 ± 0.39
Contextual Diversity [51]	Acc	38.05 ± 0.39	53.87 ± 0.38	62.36 ± 0.18	67.64 ± 0.13	70.82 ± 0.23	72.66 ± 0.12	74.46 ± 0.17	75.77 ± 0.12
	F1	31.25 ± 0.50	48.99 ± 0.29	58.77 ± 0.24	64.51 ± 0.17	68.18 ± 0.25	70.05 ± 0.11	72.13 ± 0.15	73.35 ± 0.07
Glister [52]	Acc	39.15 ± 0.23	54.57 ± 0.39	62.67 ± 0.19	67.60 ± 0.24	70.85 ± 0.27	73.07 ± 0.26	74.63 ± 0.21	76.00 ± 0.20
	F1	32.32 ± 0.31	49.72 ± 0.53	58.80 ± 0.32	64.50 ± 0.34	68.07 ± 0.38	70.47 ± 0.35	72.18 ± 0.25	73.69 ± 0.24
GraNd [53]	Acc	38.52 ± 0.06	54.65 ± 0.12	62.96 ± 0.10	67.27 ± 0.07	70.38 ± 0.07	72.56 ± 0.05	74.67 ± 0.06	75.77 ± 0.12
	F1	32.34 ± 0.10	49.89 ± 0.14	59.09 ± 0.13	64.04 ± 0.09	67.48 ± 0.09	69.81 ± 0.08	72.13 ± 0.05	73.44 ± 0.13
Forgetting [54]	Acc	38.18 ± 0.43	54.84 ± 0.23	62.55 ± 0.15	67.59 ± 0.10	70.99 ± 0.05	72.54 ± 0.07	74.81 ± 0.05	75.74 ± 0.01
	F1	31.67 ± 0.39	50.02 ± 0.20	58.64 ± 0.16	64.85 ± 0.13	68.53 ± 0.07	70.30 ± 0.05	72.59 ± 0.04	73.74 ± 0.02
DeepFool [55]	Acc	38.69 ± 0.64	54.85 ± 0.33	62.90 ± 0.21	67.77 ± 0.29	70.73 ± 0.22	73.24 ± 0.22	74.57 ± 0.23	75.71 ± 0.15
	F1	31.67 ± 0.68	49.79 ± 0.53	58.93 ± 0.32	64.42 ± 0.27	67.91 ± 0.15	70.73 ± 0.20	72.19 ± 0.29	73.39 ± 0.20
Entropy [56]	Acc	39.68 ± 0.37	54.78 ± 0.22	63.42 ± 0.18	67.95 ± 0.11	71.00 ± 0.10	73.28 ± 0.10	75.02 ± 0.08	75.82 ± 0.06
	F1	32.53 ± 0.53	49.57 ± 0.29	59.62 ± 0.25	64.55 ± 0.10	67.95 ± 0.12	70.68 ± 0.12	72.46 ± 0.12	73.29 ± 0.04
Margin [56]	Acc	39.33 ± 0.22	54.36 ± 0.17	62.66 ± 0.12	67.53 ± 0.14	71.19 ± 0.09	73.09 ± 0.14	74.66 ± 0.11	75.57 ± 0.13
	F1	32.03 ± 0.30	49.00 ± 0.23	58.62 ± 0.21	64.16 ± 0.15	68.33 ± 0.14	70.37 ± 0.17	72.03 ± 0.11	73.14 ± 0.20
Least Confidence [56]	Acc	39.00 ± 0.25	54.14 ± 0.30	63.23 ± 0.20	67.68 ± 0.11	70.99 ± 0.14	73.04 ± 0.05	74.65 ± 0.09	75.58 ± 0.08
	F1	31.83 ± 0.21	48.90 ± 0.37	59.31 ± 0.29	64.09 ± 0.20	68.03 ± 0.20	70.30 ± 0.07	72.02 ± 0.10	73.15 ± 0.12
RPVopt	Acc	**40.39 ± 0.35**	**55.48 ± 0.40**	**63.47 ± 0.30**	**68.45 ± 0.15**	**72.13 ± 0.23**	**73.72 ± 0.15**	**75.76 ± 0.24**	**76.31 ± 0.20**
	F1	**33.40 ± 0.25**	**50.35 ± 0.57**	**59.88 ± 0.39**	**65.35 ± 0.17**	**69.29 ± 0.31**	**71.45 ± 0.20**	**73.50 ± 0.28**	**73.99 ± 0.24**

We use the linear probing and two-layer finetuning as the learning problem to evaluate the performance of our method on the CIFAR10 [48] as the standard homogenous dataset and the StanfordCars [49] as the challenging non-homogeneous dataset. While the results of CIFAR10 are deferred to the appendix, Table 3 shows the results of linear probing on StanfordCars on different coreset sizes, ranging from 500 to 4000. As can be seen, our method always achieves the best test accuracy and F1 scores in all settings. Notably, for the coreset size of 500, RPVopt exceeds the second-best method by 1.4% accuracy.

Table 4: Baselines performance on StanfordCars when fine-tuning the last two layers of ResNet18.

Method	Metric	500	1000	1500	2000	2500	3000	3500	4000
Uniform Sampling	Acc	10.69 ± 0.17	18.29 ± 0.34	24.74 ± 0.36	29.19 ± 0.37	32.77 ± 0.31	35.69 ± 0.35	38.02 ± 0.31	40.35 ± 0.26
	F1	7.70 ± 0.21	15.29 ± 0.28	21.72 ± 0.34	26.14 ± 0.39	29.83 ± 0.30	32.80 ± 0.37	35.16 ± 0.30	37.51 ± 0.23
Herding [47]	Acc	11.11 ± 0.31	18.49 ± 0.45	24.53 ± 0.23	**29.19 ± 0.21**	32.42 ± 0.16	35.83 ± 0.24	38.30 ± 0.19	40.51 ± 0.19
	F1	8.06 ± 0.25	15.46 ± 0.36	21.57 ± 0.30	25.90 ± 0.24	29.48 ± 0.23	32.89 ± 0.27	35.50 ± 0.22	37.56 ± 0.21
Contextual Diversity [51]	Acc	10.30 ± 0.19	18.12 ± 0.22	24.47 ± 0.33	28.50 ± 0.34	32.66 ± 0.27	35.67 ± 0.32	38.31 ± 0.15	40.53 ± 0.18
	F1	7.66 ± 0.25	15.29 ± 0.23	21.81 ± 0.26	25.65 ± 0.40	29.79 ± 0.29	32.86 ± 0.31	35.55 ± 0.14	37.81 ± 0.23
GraNd [53]	Acc	10.72 ± 0.08	18.51 ± 0.21	24.33 ± 0.29	28.59 ± 0.17	32.67 ± 0.20	35.83 ± 0.16	**38.58 ± 0.15**	**40.70 ± 0.11**
	F1	7.82 ± 0.08	15.51 ± 0.20	21.18 ± 0.28	25.66 ± 0.15	29.70 ± 0.22	32.76 ± 0.16	**35.72 ± 0.15**	37.83 ± 0.11
Forgetting [54]	Acc	10.46 ± 0.26	18.80 ± 0.28	24.16 ± 0.21	28.61 ± 0.31	32.48 ± 0.28	35.18 ± 0.24	37.78 ± 0.22	40.24 ± 0.13
	F1	7.46 ± 0.14	15.52 ± 0.20	21.06 ± 0.23	25.64 ± 0.25	29.58 ± 0.30	32.38 ± 0.20	35.16 ± 0.18	37.41 ± 0.14
DeepFool [55]	Acc	10.65 ± 0.29	18.52 ± 0.18	**24.97 ± 0.20**	29.02 ± 0.17	32.60 ± 0.18	35.59 ± 0.24	38.20 ± 0.22	39.98 ± 0.35
	F1	7.89 ± 0.18	15.44 ± 0.23	**22.11 ± 0.11**	26.08 ± 0.29	29.83 ± 0.27	32.92 ± 0.33	35.47 ± 0.22	37.28 ± 0.40
Entropy [56]	Acc	10.30 ± 0.07	18.48 ± 0.13	24.25 ± 0.26	28.87 ± 0.13	32.84 ± 0.20	35.64 ± 0.20	37.96 ± 0.11	40.29 ± 0.27
	F1	7.69 ± 0.11	15.31 ± 0.23	21.24 ± 0.24	25.95 ± 0.17	30.03 ± 0.17	32.85 ± 0.23	35.19 ± 0.12	37.33 ± 0.34
Margin [56]	Acc	10.58 ± 0.32	18.37 ± 0.26	24.36 ± 0.19	29.18 ± 0.12	32.73 ± 0.15	35.67 ± 0.30	38.27 ± 0.20	40.58 ± 0.06
	F1	7.93 ± 0.22	15.41 ± 0.19	21.33 ± 0.22	26.15 ± 0.12	29.66 ± 0.05	32.86 ± 0.30	35.61 ± 0.17	37.77 ± 0.07
LeastConfidence [56]	Acc	10.64 ± 0.23	18.45 ± 0.30	24.72 ± 0.20	29.05 ± 0.07	**32.88 ± 0.13**	35.66 ± 0.18	38.25 ± 0.20	39.91 ± 0.09
	F1	7.80 ± 0.10	15.47 ± 0.37	21.75 ± 0.25	26.18 ± 0.04	30.03 ± 0.14	32.79 ± 0.15	35.42 ± 0.16	37.14 ± 0.12
RPVopt	Acc	**11.12 ± 0.21**	**19.11 ± 0.24**	24.82 ± 0.15	29.13 ± 0.27	32.70 ± 0.19	**36.05 ± 0.27**	38.57 ± 0.12	40.56 ± 0.24
	F1	**8.18 ± 0.16**	**16.19 ± 0.23**	22.09 ± 0.21	**26.43 ± 0.31**	**30.33 ± 0.23**	**33.32 ± 0.27**	35.68 ± 0.15	**37.87 ± 0.22**

We also evaluate the performance of comparative methods finetuning the last two-layer of ResNet18 [42]. In Table 4, we showcase the results on StanfordCars, where RPVopt demonstrates impressive efficiency, boosting the performance of uniform sampling across different setups.

References

[1] 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.

[2] 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.

[3] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[4] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[5] Alec Radford. Improving language understanding by generative pre-training. 2018.

[6] 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.

[7] Madeleine Udell and Alex Townsend. Why are big data matrices approximately low rank? *SIAM Journal on Mathematics of Data Science*, 1(1):144–160, 2019.

[8] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. *arXiv preprint arXiv:2012.13255*, 2020.

[9] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[10] Yao Xiao, Lu Xu, Jiaxi Li, Wei Lu, and Xiaoli Li. Decomposed prompt tuning via low-rank reparameterization. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 13335–13347, 2023.

[11] Sanae Lotfi, Marc Finzi, Sanyam Kapoor, Andres Potapczynski, Micah Goldblum, and Andrew G Wilson. Pac-bayes compression bounds so tight that they can explain generalization. *Advances in Neural Information Processing Systems*, 35:31459–31473, 2022.

[12] Kathryn Chaloner and Isabella Verdinelli. Bayesian experimental design: A review. *Statistical science*, pages 273–304, 1995.

[13] Friedrich Pukelsheim. *Optimal design of experiments*. SIAM, 2006.

[14] Valerii Vadimovich Fedorov. *Theory of optimal experiments*. Elsevier, 2013.

[15] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[16] Yijun Dong, Hoang Phan, Xiang Pan, and Qi Lei. Sketchy moment matching: Toward fast and provable data selection for finetuning. *Advances in Neural Information Processing Systems*, 2024.

[17] Zeyuan Allen-Zhu, Yuanzhi Li, Aarti Singh, and Yining Wang. Near-optimal discrete optimization for experimental design: A regret minimization approach. *arXiv preprint arXiv:1711.05174*, 2017.

[18] Yining Wang, Adams Wei Yu, and Aarti Singh. On computationally tractable selection of experiments in measurement-constrained regression models. *Journal of Machine Learning Research*, 18(143):1–41, 2017.

[19] Kai Yu, Jinbo Bi, and Volker Tresp. Active learning via transductive experimental design. In *Proceedings of the 23rd international conference on Machine learning*, pages 1081–1088, 2006.

[20] Neta Shoham and Haim Avron. Experimental design for overparameterized learning with application to single shot deep active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10):11766–11777, 2023.

[21] Zeyuan Allen-Zhu, Yuanzhi Li, Aarti Singh, and Yining Wang. Near-optimal discrete optimization for experimental design: A regret minimization approach. *Mathematical Programming*, 186:439–478, 2021.

[22] Lap Chi Lau and Hong Zhou. A local search framework for experimental design. *SIAM Journal on Computing*, 51(4):900–951, 2022.

[23] Haim Avron and Christos Boutsidis. Faster subset selection for matrices and applications. *idea*, 10(2):2, 2012.

[24] FR De Hoog and RMM Mattheij. Subset selection for matrices. *Linear Algebra and its Applications*, 422(2-3):349–359, 2007.

[25] Alex Gittens. The spectral norm error of the naive nystrom extension. *arXiv preprint arXiv:1110.5305*, 2011.

[26] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, 43(2):687–717, 2014.

[27] Nathan Halko, Per-Gunnar Martinsson, and Joel A Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[28] David P Woodruff et al. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157, 2014.

[29] Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*, 2024.

[30] Yifan Chen, Ethan N Epperly, Joel A Tropp, and Robert J Webber. Randomly pivoted cholesky: Practical approximation of a kernel matrix with few entry evaluations. *arXiv preprint arXiv:2207.06503*, 2022.

[31] Yijun Dong, Chao Chen, Per-Gunnar Martinsson, and Katherine Pearce. Robust blockwise random pivoting: Fast and accurate adaptive interpolative decomposition. *arXiv preprint arXiv:2309.16002*, 2023.

[32] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

[33] Nicholas J Higham. *Accuracy and stability of numerical algorithms*. SIAM, 2002.

[34] Jelani Nelson and Huy L Nguyên. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *2013 ieee 54th annual symposium on foundations of computer science*, pages 117–126. IEEE, 2013.

[35] Jean Bourgain, Sjoerd Dirksen, and Jelani Nelson. Toward a unified theory of sparse dimensionality reduction in euclidean space. In *Proceedings of the forty-seventh annual ACM symposium on Theory of Computing*, pages 499–508, 2015.

[36] Michael B Cohen. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 278–287. SIAM, 2016.

[37] Per-Gunnar Martinsson and Joel A Tropp. Randomized numerical linear algebra: Foundations and algorithms. *Acta Numerica*, 29:403–572, 2020.

[38] Mu Li, Gary L Miller, and Richard Peng. Iterative row sampling. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 127–136. IEEE, 2013.

[39] Ahmed Alaoui and Michael W Mahoney. Fast randomized kernel ridge regression with statistical guarantees. *Advances in neural information processing systems*, 28, 2015.

[40] Michael B Cohen, Cameron Musco, and Christopher Musco. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1758–1777. SIAM, 2017.

[41] Zhifei Zhang, Yang Song, and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5810–5818, 2017.

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

[43] Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[44] Amit Deshpande and Santosh Vempala. Adaptive sampling and fast low-rank matrix approximation. In *International Workshop on Approximation Algorithms for Combinatorial Optimization*, pages 292–303. Springer, 2006.

[45] Samprit Chatterjee and Ali S Hadi. Influential observations, high leverage points, and outliers in linear regression. *Statistical science*, pages 379–393, 1986.

[46] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.

[47] Max Welling. Herding dynamical weights to learn. In *Proceedings of the 26th annual international conference on machine learning*, pages 1121–1128, 2009.

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

[49] Jonathan Krause, Jia Deng, Michael Stark, and Li Fei-Fei. Collecting a large-scale dataset of fine-grained cars. 2013.

[50] Chengcheng Guo, Bo Zhao, and Yanbing Bai. Deepcore: A comprehensive library for coreset selection in deep learning. In *International Conference on Database and Expert Systems Applications*, pages 181–195. Springer, 2022.

[51] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. Contextual diversity for active learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pages 137–153. Springer, 2020.

[52] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. Glister: Generalization based data subset selection for efficient and robust learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8110–8118, 2021.

[53] 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.

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

[55] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[56] 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. *arXiv preprint arXiv:1906.11829*, 2019.

[57] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-gan: Speeding up gan training using core-sets. In *International Conference on Machine Learning*, pages 9005–9015. PMLR, 2020.

[58] Yutian Chen, Max Welling, and Alex Smola. Super-samples from kernel herding. In *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*, pages 109–116, 2010.

[59] Tamas Sarlos. Improved approximation algorithms for large matrices via random projections. In *2006 47th annual IEEE symposium on foundations of computer science (FOCS'06)*, pages 143–152. IEEE, 2006.

[60] Kyriakos Axiotis, Vincent Cohen-Addad, Monika Henzinger, Sammy Jerome, Vahab Mirrokni, David Saulpic, David Woodruff, and Michael Wunder. Data-efficient learning via clustering-based sensitivity sampling: Foundation models and beyond. *arXiv preprint arXiv:2402.17327*, 2024.

[61] Xue Chen and Eric Price. Active regression via linear-sample sparsification. In *Conference on Learning Theory*, pages 663–695. PMLR, 2019.

[62] Atsushi Shimizu, Xiaoou Cheng, Christopher Musco, and Jonathan Weare. Improved active learning via dependent leverage score sampling. *arXiv preprint arXiv:2310.04966*, 2023.

[63] Aarshvi Gajjar, Wai Ming Tai, Xu Xingyu, Chinmay Hegde, Christopher Musco, and Yi Li. Agnostic active learning of single index models with linear sample complexity. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 1715–1754. PMLR, 2024.

[64] Amit Deshpande, Luis Rademacher, Santosh S Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. *Theory of Computing*, 2(1):225–247, 2006.

[65] Michal Derezinski, Rajiv Khanna, and Michael W Mahoney. Improved guarantees and a multiple-descent curve for column subset selection and the nystrom method. *Advances in Neural Information Processing Systems*, 33:4953–4964, 2020.

[66] Yijun Dong and Per-Gunnar Martinsson. Simpler is better: a comparative study of randomized pivoting algorithms for cur and interpolative decompositions. *Advances in Computational Mathematics*, 49(4):66, 2023.

[67] Ethan N Epperly, Joel A Tropp, and Robert J Webber. Embrace rejection: Kernel matrix approximation by accelerated randomly pivoted cholesky. *arXiv preprint arXiv:2410.03969*, 2024.

[68] Yiping Wang, Yifang Chen, Wendan Yan, Kevin Jamieson, and Simon Shaolei Du. Variance alignment score: A simple but tough-to-beat data selection method for multimodal contrastive learning. *arXiv preprint arXiv:2402.02055*, 2024.

[69] Artem Vysogorets, Kartik Ahuja, and Julia Kempe. Robust data pruning: Uncovering and overcoming implicit bias. *arXiv preprint arXiv:2404.05579*, 2024.

A Additional Discussions

A.1 Additional Related Works

Data selection. Recent data selection methods identify important samples through training dynamics [54, 53], yielding additional annotation and computational costs, unaligned with the original goal of selecting data to reduce training costs. Label-free alternatives evaluate the importance of data via geometric properties in the embedding space [57, 51, 58]. These methods remove redundancy to form diverse, representative coresets without extensive labeling or (early-stage) training. This underlying idea of data selection based on geometric information is closely related to various widely studied problems like coreset selection [59, 28, 39, 60], active learning [61, 62, 63], and matrix skeletonization [64, 65, 66, 30, 31, 67]. More recent work named Variance Alignment Score (VAS) [68] and Sketchy Moment Matching [16] aligns some high-level statistics of the selected samples with that of the original data distribution, and additionally enhances model performance through data filtering or gradient sketching ideas respectively. These advances highlight the value of selecting informative and diverse coreset, especially in complex tasks.

A.2 Limitations and Future Directions

In this work, we introduce a fast data selection algorithm, RPVopt, for high-dimensional learning problems with low intrinsic dimensions. Concretely, RPVopt leverages the data selection criterion proposed in [16], first exploring the low intrinsic dimension in data via sketching, and then exploiting the information in the resulting low-dimensional subspace by adaptively sampling data that optimize the selection criterion locally. The appealing empirical performance of RPVopt is demonstrated in both regression and classification settings. A natural question regarding the theoretical guarantee for RPVopt remains open and is a work in progress.

Beyond the theoretical guarantee, the potentials and limitations of RPVopt are not fully understood in the current stage. First, while RPVopt is inspired by the analysis for regression problems, its competitive performance extends to the classification setting in practice. Second, the strong performance of RPVopt for unsupervised data selection on imbalanced classification tasks (*e.g.*, StanfordCars) suggests its potential in the context of distributionally robust data selection [69], beyond the classical statistical learning setting under the *i.i.d.* sampling assumption. Understanding the mechanism behind RPVopt in more comprehensive settings like classification and distributionally robust learning is an exciting future direction.

B Experiment Details and Additional Results

Due to the space constraint, some details were omitted in the main paper. We here provide the detailed training configuration in Section B.1 and additional experiment results in Section B.2.

B.1 Hyperparameter Selection and Training Details

We sweep the sketching dimension $m \in \{32, 64, 128, 256, 512\}$, the block size $b \in \{5, 10, 15, 20\}$ and temperature $\tau = e^{-3}$ for all experiments. We use the feature before the last linear layer in linear probing and the last-two layer gradients in two-layer finetuning to perform sketching and selection. For linear probing, we train the model 50 epochs with Adam optimizer, learning rate 1e-1 and batch size 512, for two-layer finetuning, we use the learning rate 10^{-2} and batch size 512.

B.2 Additional Experimental Results

Table 5 showcases the effectiveness of RPVopt on the CIFAR10 dataset with CLIP backbone. Overall, RPVopt achieves superior performance in terms of F1 and accuracy scores compared to other baselines. Notably, at the smallest coreset size, our proposed method outperforms other baselines by large margins. Across all setups, RPVopt either achieves the best performance or is comparable to the top-performing method, further confirming the robustness and effectiveness of RPVopt.

Table 5: CIFAR10 with CLIP Linear Probing. Best results are highlighted in **bold**, standard errors with 5 random seeds. Different columns indicate different core set sizes.

Method	Metric	50	100	200	500	1000	1500	2000	2500	3000	3500	4000
Uniform	Acc	79.82 ± 2.06	88.51 ± 0.44	90.90 ± 0.13	**92.47 ± 0.12**	**93.00 ± 0.07**	93.23 ± 0.08	93.39 ± 0.06	93.57 ± 0.03	**93.68 ± 0.05**	93.72 ± 0.05	**93.80 ± 0.04**
	F1	78.41 ± 2.72	88.56 ± 0.45	90.92 ± 0.12	**92.47 ± 0.12**	**93.00 ± 0.07**	93.24 ± 0.08	93.40 ± 0.06	93.58 ± 0.03	**93.69 ± 0.05**	93.73 ± 0.05	**93.82 ± 0.04**
Herding [47]	Acc	86.09 ± 0.34	88.21 ± 0.54	90.77 ± 0.15	91.81 ± 0.20	92.73 ± 0.08	93.10 ± 0.06	93.34 ± 0.02	93.34 ± 0.08	93.60 ± 0.07	93.76 ± 0.04	93.67 ± 0.06
	F1	86.29 ± 0.33	88.36 ± 0.52	90.81 ± 0.14	91.83 ± 0.19	92.74 ± 0.08	93.11 ± 0.06	93.35 ± 0.02	93.35 ± 0.08	93.61 ± 0.07	93.78 ± 0.04	93.69 ± 0.06
Contextual Diversity [51]	Acc	82.76 ± 1.00	88.96 ± 0.34	90.98 ± 0.25	92.10 ± 0.19	92.69 ± 0.10	93.09 ± 0.07	93.30 ± 0.06	93.46 ± 0.04	93.58 ± 0.03	93.61 ± 0.03	93.77 ± 0.06
	F1	82.68 ± 1.16	89.01 ± 0.34	91.01 ± 0.24	92.12 ± 0.18	92.70 ± 0.10	93.09 ± 0.07	93.30 ± 0.06	93.45 ± 0.04	93.58 ± 0.03	93.61 ± 0.03	93.77 ± 0.06
Glister [52]	Acc	82.19 ± 2.00	88.37 ± 0.36	90.96 ± 0.30	92.41 ± 0.20	92.97 ± 0.10	**93.30 ± 0.07**	93.42 ± 0.09	93.58 ± 0.06	93.66 ± 0.06	93.70 ± 0.03	93.80 ± 0.04
	F1	81.79 ± 2.33	88.45 ± 0.37	91.01 ± 0.29	92.41 ± 0.19	92.96 ± 0.10	**93.31 ± 0.07**	93.42 ± 0.09	93.59 ± 0.07	93.68 ± 0.06	93.71 ± 0.03	93.80 ± 0.05
GraNd [53]	Acc	82.44 ± 0.30	88.98 ± 0.04	90.38 ± 0.02	92.50 ± 0.03	92.90 ± 0.02	93.24 ± 0.02	93.38 ± 0.02	93.56 ± 0.02	93.56 ± 0.02	93.70 ± 0.03	93.77 ± 0.03
	F1	82.30 ± 0.33	88.87 ± 0.04	90.38 ± 0.03	92.49 ± 0.03	92.92 ± 0.02	93.26 ± 0.02	93.39 ± 0.02	93.57 ± 0.02	93.57 ± 0.02	93.71 ± 0.03	93.79 ± 0.03
Forgetting [54]	Acc	82.16 ± 0.80	87.84 ± 0.67	90.28 ± 0.23	91.67 ± 0.16	92.90 ± 0.06	93.09 ± 0.06	93.23 ± 0.05	93.41 ± 0.04	93.50 ± 0.05	93.62 ± 0.02	93.70 ± 0.05
	F1	82.02 ± 0.92	87.90 ± 0.65	90.30 ± 0.22	91.67 ± 0.16	92.90 ± 0.06	93.08 ± 0.05	93.23 ± 0.05	93.40 ± 0.04	93.50 ± 0.05	93.62 ± 0.02	93.71 ± 0.05
DeepFool [55]	Acc	81.50 ± 1.20	88.15 ± 0.33	90.89 ± 0.16	92.20 ± 0.11	93.07 ± 0.06	93.29 ± 0.06	93.39 ± 0.04	93.61 ± 0.04	93.65 ± 0.03	93.75 ± 0.04	93.79 ± 0.05
	F1	80.91 ± 1.48	88.08 ± 0.32	90.87 ± 0.18	92.19 ± 0.11	93.07 ± 0.06	93.29 ± 0.06	93.39 ± 0.04	93.61 ± 0.04	93.65 ± 0.03	93.75 ± 0.05	93.80 ± 0.05
Entropy [56]	Acc	76.86 ± 0.26	82.32 ± 0.54	90.02 ± 0.05	92.08 ± 0.09	92.99 ± 0.02	93.09 ± 0.04	93.28 ± 0.04	93.42 ± 0.02	93.51 ± 0.04	93.62 ± 0.03	93.70 ± 0.01
	F1	75.19 ± 0.32	81.64 ± 0.67	90.07 ± 0.04	92.10 ± 0.08	92.99 ± 0.02	93.10 ± 0.04	93.28 ± 0.04	93.43 ± 0.02	93.52 ± 0.04	93.63 ± 0.04	93.71 ± 0.01
Margin [56]	Acc	77.06 ± 0.52	87.08 ± 0.48	89.35 ± 0.12	92.11 ± 0.04	92.95 ± 0.05	93.13 ± 0.03	93.27 ± 0.03	93.48 ± 0.06	93.47 ± 0.04	93.59 ± 0.03	93.73 ± 0.04
	F1	76.05 ± 0.58	87.10 ± 0.52	89.43 ± 0.12	92.13 ± 0.04	92.97 ± 0.05	93.13 ± 0.03	93.28 ± 0.03	93.49 ± 0.06	93.48 ± 0.04	93.60 ± 0.03	93.74 ± 0.04
LeastConfidence [56]	Acc	77.35 ± 0.18	84.46 ± 0.15	90.27 ± 0.20	92.09 ± 0.06	92.85 ± 0.04	93.27 ± 0.03	93.44 ± 0.03	93.56 ± 0.04	93.60 ± 0.03	93.57 ± 0.02	93.58 ± 0.04
	F1	76.24 ± 0.19	84.42 ± 0.16	90.31 ± 0.19	92.11 ± 0.06	92.86 ± 0.04	93.27 ± 0.03	93.44 ± 0.03	93.58 ± 0.04	93.61 ± 0.03	93.58 ± 0.03	93.59 ± 0.04
RPVopt	Acc	**86.09 ± 0.34**	**88.98 ± 0.04**	**90.96 ± 0.30**	92.41 ± 0.20	92.97 ± 0.10	93.24 ± 0.02	**93.44 ± 0.03**	**93.61 ± 0.04**	93.66 ± 0.06	**93.75 ± 0.04**	**93.80 ± 0.04**
	F1	**86.29 ± 0.33**	**88.87 ± 0.04**	**91.01 ± 0.29**	92.41 ± 0.19	92.96 ± 0.10	93.26 ± 0.02	**93.44 ± 0.03**	**93.61 ± 0.04**	93.68 ± 0.06	**93.75 ± 0.05**	93.80 ± 0.05

Table 6: Performance of baselines on CIFAR10 when finetuning the last two layers of ResNet18.

Method	Metric	50	100	200	500	1000	1500	2000	2500	3000	3500	4000
Uniform Sampling	Acc	44.56 ± 0.85	55.30 ± 0.50	62.90 ± 0.42	70.56 ± 0.24	74.18 ± 0.23	75.90 ± 0.15	76.90 ± 0.21	77.83 ± 0.09	**78.39 ± 0.12**	78.29 ± 0.20	**78.92 ± 0.06**
	F1	40.82 ± 1.74	53.60 ± 0.69	62.38 ± 0.49	70.44 ± 0.28	74.12 ± 0.25	75.80 ± 0.18	**76.83 ± 0.23**	77.76 ± 0.10	**78.34 ± 0.11**	78.25 ± 0.20	**78.85 ± 0.05**
Herding [47]	Acc	40.20 ± 1.75	53.06 ± 1.46	62.35 ± 0.35	70.25 ± 0.25	73.96 ± 0.22	75.59 ± 0.10	76.46 ± 0.20	77.40 ± 0.20	77.73 ± 0.13	78.27 ± 0.20	78.57 ± 0.09
	F1	35.68 ± 2.39	51.34 ± 1.87	61.79 ± 0.31	69.93 ± 0.40	73.83 ± 0.23	75.46 ± 0.09	76.37 ± 0.21	77.27 ± 0.21	77.67 ± 0.12	78.22 ± 0.18	78.52 ± 0.11
Contextual Diversity [51]	Acc	46.09 ± 1.07	54.57 ± 0.78	62.21 ± 0.27	70.54 ± 0.36	74.36 ± 0.29	75.75 ± 0.18	76.79 ± 0.15	77.34 ± 0.15	77.90 ± 0.13	78.48 ± 0.13	78.69 ± 0.17
	F1	42.55 ± 1.14	52.99 ± 0.88	61.51 ± 0.48	70.27 ± 0.35	74.24 ± 0.33	75.67 ± 0.19	76.71 ± 0.15	77.19 ± 0.14	77.75 ± 0.14	78.38 ± 0.15	78.57 ± 0.16
Glister [52]	Acc	44.33 ± 1.71	53.32 ± 1.13	62.15 ± 0.65	69.77 ± 0.20	74.05 ± 0.20	75.63 ± 0.19	76.60 ± 0.09	77.48 ± 0.14	77.97 ± 0.07	78.22 ± 0.16	78.32 ± 0.15
	F1	41.92 ± 2.16	51.81 ± 1.29	61.80 ± 0.61	69.64 ± 0.19	73.95 ± 0.20	75.55 ± 0.20	76.52 ± 0.08	77.40 ± 0.12	77.89 ± 0.08	78.18 ± 0.16	78.21 ± 0.15
GraNd [53]	Acc	42.73 ± 0.76	53.58 ± 1.14	63.30 ± 0.54	70.30 ± 0.03	74.03 ± 0.10	75.40 ± 0.03	76.42 ± 0.08	77.32 ± 0.26	77.79 ± 0.13	77.94 ± 0.13	78.50 ± 0.16
	F1	39.02 ± 1.15	51.82 ± 1.63	63.04 ± 0.57	70.17 ± 0.07	73.90 ± 0.11	75.30 ± 0.03	76.34 ± 0.08	77.28 ± 0.27	77.74 ± 0.14	77.93 ± 0.10	78.40 ± 0.16
Forgetting [54]	Acc	43.16 ± 1.78	53.58 ± 1.48	63.07 ± 0.45	70.75 ± 0.23	73.74 ± 0.16	75.25 ± 0.26	76.24 ± 0.12	77.36 ± 0.10	77.92 ± 0.18	78.07 ± 0.12	78.41 ± 0.12
	F1	40.55 ± 1.26	53.06 ± 1.59	62.47 ± 0.38	70.55 ± 0.28	73.60 ± 0.16	75.13 ± 0.26	76.10 ± 0.08	77.26 ± 0.15	77.85 ± 0.17	77.95 ± 0.10	78.31 ± 0.08
DeepFool [55]	Acc	43.88 ± 0.46	55.51 ± 0.65	63.05 ± 0.64	70.53 ± 0.27	73.85 ± 0.16	75.60 ± 0.18	76.54 ± 0.17	77.21 ± 0.17	77.77 ± 0.13	77.99 ± 0.13	78.39 ± 0.10
	F1	40.64 ± 0.99	54.07 ± 0.13	62.62 ± 0.66	70.33 ± 0.31	73.74 ± 0.19	75.50 ± 0.19	76.41 ± 0.16	77.10 ± 0.17	77.69 ± 0.13	77.91 ± 0.15	78.29 ± 0.11
Entropy [56]	Acc	44.49 ± 1.03	54.20 ± 1.47	63.72 ± 0.40	70.61 ± 0.08	74.47 ± 0.14	75.66 ± 0.06	76.66 ± 0.15	77.53 ± 0.32	78.09 ± 0.14	78.22 ± 0.02	78.60 ± 0.13
	F1	41.64 ± 1.36	52.14 ± 1.41	63.24 ± 0.44	70.30 ± 0.20	74.34 ± 0.16	75.55 ± 0.05	76.57 ± 0.14	77.47 ± 0.32	78.04 ± 0.15	78.17 ± 0.02	78.52 ± 0.13
Margin [56]	Acc	43.67 ± 0.55	53.50 ± 1.70	62.67 ± 0.44	70.80 ± 0.29	**74.75 ± 0.16**	75.71 ± 0.14	76.58 ± 0.16	77.71 ± 0.17	78.11 ± 0.22	78.25 ± 0.12	78.63 ± 0.22
	F1	41.37 ± 1.29	51.50 ± 1.93	62.12 ± 0.34	70.49 ± 0.35	**74.64 ± 0.18**	75.59 ± 0.13	76.49 ± 0.14	77.67 ± 0.18	78.02 ± 0.23	78.13 ± 0.15	78.56 ± 0.21
LeastConfidence [56]	Acc	44.60 ± 1.74	53.94 ± 0.70	62.56 ± 0.53	**70.91 ± 0.20**	74.55 ± 0.12	**76.28 ± 0.10**	76.68 ± 0.11	77.54 ± 0.23	78.04 ± 0.19	78.30 ± 0.18	78.78 ± 0.15
	F1	40.55 ± 2.19	52.51 ± 1.01	61.97 ± 0.65	**70.69 ± 0.18**	74.27 ± 0.26	**76.13 ± 0.10**	76.63 ± 0.09	77.43 ± 0.22	77.98 ± 0.21	78.19 ± 0.19	78.73 ± 0.16
RPVopt	Acc	**45.92 ± 1.30**	**55.82 ± 0.92**	**63.52 ± 0.81**	70.75 ± 0.26	74.53 ± 0.16	76.03 ± 0.29	**76.92 ± 0.12**	**77.90 ± 0.17**	78.19 ± 0.10	**78.58 ± 0.09**	78.88 ± 0.14
	F1	**43.00 ± 1.68**	**54.33 ± 1.08**	**63.08 ± 1.02**	70.63 ± 0.26	74.38 ± 0.17	75.89 ± 0.30	76.78 ± 0.15	**77.76 ± 0.17**	78.03 ± 0.11	**78.46 ± 0.09**	78.80 ± 0.13

Similar to the experiment in the main paper, we finetune ResNet18 by updating the weights of the last two layers only and provide the obtained scores in Table 6. Throughout our experiments, RPVopt demonstrates competitive performance, especially with large improvements observed on non-homogeneous datasets such as StanfordCars. When the sample size is small, e.g. 50 in CIFAR10 and 500 in StanfordCars, our method effectively identifies and selects informative samples that benefit downstream tasks.