Uniform Prototype Selection via Partial Optimal Transport with Submodular Guarantees

Prateek Chanda[1*†] **Prayas Agrawal**[3*] **Karthik S. Gurumoorthy**[4]

Ganesh Ramakrishnan[1,2] **Bamdev Mishra**[5] **Pratik Jawanpuria**[2]

[1] Department of Computer Science and Engineering, Indian Institute of Technology Bombay
[2] Centre for Machine Intelligence and Data Science, Indian Institute of Technology Bombay
[3] Microsoft Research India [4] Walmart Global Tech, India [5] Microsoft India

Abstract

Selecting prototypical examples from a source distribution to represent a target data distribution is a fundamental problem in machine learning. Existing subset selection methods often rely on implicit importance scores, which can be skewed towards majority classes and lead to low-quality prototypes for minority classes. We present `UniPROT`, a novel subset selection framework that minimizes the optimal transport (OT) distance between a uniformly weighted prototypical distribution and the target distribution. While intuitive, this formulation leads to a cardinality-constrained maximization of a *super-additive* objective, which is generally intractable to approximate efficiently. To address this, we propose a principled reformulation of the OT marginal constraints, yielding a partial optimal transport-based submodular objective. We prove that this reformulation enables a greedy algorithm with a $(1 - 1/e)$ approximation guarantee relative to the original super-additive maximization problem. Empirically, we showcase that enforcing uniform prototype weights in `UniPROT` consistently improves minority-class representation in imbalanced classification benchmarks without compromising majority-class accuracy. In both finetuning and pretraining regimes for large language models under domain imbalance, `UniPROT` enforces uniform source contributions, yielding robust performance gains. Our results establish `UniPROT` as a scalable, theoretically grounded solution for uniform-weighted prototype selection. Our code is publicly available at GitHub[1]

1 INTRODUCTION

Prototype selection is a fundamental problem in representation learning. The goal is to identify a subset of representative elements from a source set or distribution that faithfully summarizes a given target set or distribution. In cases where the source and target sets coincide, the problem reduces to the well-known medoids selection task. Prototypical examples have proven useful in a variety of applications, including domain understanding via summarization (Schlegel et al., 2017; Chen et al., 2019), identifying anomalies (Kawano et al., 2022), positive-unlabeled learning (Dhurandhar and Gurumoorthy, 2020; Riaz et al., 2023), and efficient training of deep models (Mirzasoleiman et al., 2020a; Killamsetty et al., 2021a; Kothawade et al., 2021; Zheng et al., 2023; Liu et al., 2024; Tan et al., 2025).

Recent prototype selection algorithms usually select a prototypical set of size k whose underlying distribution is *closest* to the target distribution, as measured by a chosen divergence or distance metric between probability distributions. Prior works (Kim et al., 2016; Gurumoorthy et al., 2019, 2021) have explored metrics such as maximum mean discrepancy (MMD) and optimal transport (OT) distance (Wasserstein distance) to quantify their closeness. Interestingly, classical problems like submodular facility location (Lin and Bilmes, 2011; Krause and Golovin, 2014) or k-medoids problems may be viewed as special cases of OT based prototype selection. Submodular optimization has been widely adopted in this context due to its favorable approximation guarantees and scalability.

Prototype selection algorithms often yield a *weighted* subset of representative instances, where the weights reflect the

*Equal contribution.
†Correspondence to: `prateekch@cse.iitb.ac.in`.

Proceedings of the 29th International Conference on Artificial Intelligence and Statistics (AISTATS) 2026, Tangier, Morocco. PMLR: Volume 300.

[1] Code: https://github.com/efficiency-learning/UniPROT

relative importance of each exemplar and are typically inferred implicitly during the selection process. However, for interpretability, *uniform weighting* is generally preferred, as disproportionate emphasis can bias human perception (Solso et al., 2017). This issue is particularly pronounced in long tailed class distributions, where minority classes may receive lower-weighted prototypes, leading to unfair or under-representative selections. Therefore, it is beneficial to design methods that promote equal importance among selected prototypes. This ensures balanced and interpretable representation, particularly in long-tailed settings.

In this work, we focus on the problem of selecting a uniformly weighted prototypical set which is closest to the target set under the optimal transport metric. We begin by showing that popular formulations such as submodular facility location and k-medoids inherently produce weighted prototype sets when viewed through the lens of OT. Motivated by this observation, we propose a novel subset selection problem aimed at identifying an equally weighted prototypical set. Although intuitively appealing, the proposed formulation corresponds to a monotone, non-negative, super-additive maximization problem, which is not directly amenable to greedy optimization. To address this challenge, we design a tight, monotone, non-negative, *submodular surrogate objective* that approximates the original super-additive problem. This reformulation enables the use of greedy algorithms with strong approximation guarantees. Furthermore, we prove that the same theoretical guarantees hold for the original super-additive maximization problem, thereby validating the effectiveness of our approach.

Our main contributions are summarized as follows:

- We formalize uniform prototype selection as a super-additive maximization problem under the cardinality constraint of selecting k prototypical examples from a source set and introduce a submodular reformulation with provable guarantees.
- We show that the proposed problem corresponds to a monotone, non-negative, super-additive maximization problem under cardinality constraint. To the best of our knowledge, efficient algorithms with provable guarantees are not known for this class.
- We prove that our submodular reformulation is equivalent to the original super-additive problem with cardinality constraint k, using which we establish a $(1 - 1/e)$ approximation guarantee for the latter. We also develop an efficient greedy algorithm whose computational cost is comparable to that of solving the submodular k-medoids problem.
- We demonstrate the utility of selecting uniformly weighted prototypical set in applications such as long tailed image classification and high quality mini-batch selection for large language model (LLM) training. Our proposed method, **`UniPROT`**, outperforms existing prototype selection methods across various benchmark datasets both in terms of solution quality and computational efficiency.

2 PRELIMINARIES

Let $\mathcal{S} := \{\mathbf{x}_i\}_{i=1}^m$ and $\mathcal{T} := \{\mathbf{y}_j\}_{j=1}^n$ be the source and the target datasets, respectively, where $\mathbf{x}_i \in \mathcal{X}$ and $\mathbf{y}_j \in \mathcal{Y}$. The corresponding source and target empirical distributions may be written as $\mu = \sum_{i=1}^m \boldsymbol{\mu}_i \delta_{\mathbf{x}_i}$ and $\nu = \sum_{j=1}^n \boldsymbol{\nu}_j \delta_{\mathbf{y}_j}$ where $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ denote the mass associated with $\mathbf{x}_i$ and $\mathbf{y}_j$, respectively, and $\delta_{\mathbf{z}}$ denote the Dirac measure centered at $\mathbf{z}$. If μ and ν are probability distributions, $\boldsymbol{\mu} \in \Delta_m$ and $\boldsymbol{\nu} \in \Delta_n$ where $\Delta_m := \{\mathbf{z} \in \mathbb{R}_+^m \mid \mathbf{z}^\top \mathbf{1} = 1\}$ and $\mathbf{1}$ denote the vector of ones of appropriate size. For $\mathbf{z} \in \mathbb{R}_+^m$, let $\mathrm{supp}(\mathbf{z}) = \{i \in [m] \mid \mathbf{z}_i > 0\}$. For any subset $\mathcal{P} \subseteq \mathcal{S}$, let $\mathcal{I}_\mathcal{P}$ be the set of indices corresponding to the points $\mathbf{x} \in \mathcal{P}$, i.e. $\mathcal{I}_\mathcal{P} = \{i : \mathbf{x}_i \in \mathcal{P}\}$. Let $\mathbf{Z}(\mathcal{I}_\mathcal{P}, :)$ denote the sub-matrix of $\mathbf{Z}$ containing rows corresponding to the indices in $\mathcal{I}_\mathcal{P}$, and when $\mathcal{I}_\mathcal{P} = \{i\}$ is a singleton set, we represent the i-th row the matrix $\mathbf{Z}$ as $\mathbf{Z}(i, :)$. Lastly, let $[m] = \{1, 2, \ldots, m\}$ for $m \in \mathbb{N}$.

Optimal Transport (OT) problem (Kantorovich, 1942; Peyré et al., 2019) seeks a transport plan $\boldsymbol{\gamma}$ that minimizes the total cost of moving mass from a source distribution μ to a target distribution ν:

$$\mathrm{OT}_{\min}(\boldsymbol{\mu}, \boldsymbol{\nu}) = \min_{\boldsymbol{\gamma} \in \Gamma(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \mathbf{C}, \boldsymbol{\gamma} \rangle, \tag{1}$$

where $\boldsymbol{\mu} \in \Delta_m, \boldsymbol{\nu} \in \Delta_n, \Gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) = \{\boldsymbol{\gamma} \in \mathbb{R}_+^{m \times n} \mid \boldsymbol{\gamma}\mathbf{1} = \boldsymbol{\mu}, \boldsymbol{\gamma}^\top \mathbf{1} = \boldsymbol{\nu}\}$ is the set of admissible couplings and $\mathbf{C} \in \mathbb{R}^{m \times n}$ denote a cost matrix induced by a ground cost function $c : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+ : (\mathbf{x}, \mathbf{y}) \mapsto c(\mathbf{x}, \mathbf{y})$ such that $\mathbf{C}_{ij} = c(\mathbf{x}_i, \mathbf{y}_j)$. Hence, $\mathbf{C}_{ij}$ is the cost of transport a unit mass from $\mathbf{x}_i$ to $\mathbf{y}_j$. When c is a distance (e.g., ℓ_1 or ℓ_2 distance), OT cost induces the Wasserstein distance between the probability distributions $\boldsymbol{\mu}$ and $\boldsymbol{\nu}$. In doing so, OT lifts the geometry from the underlying sample space to the space of probability measures, enabling a rich geometric framework for comparing distributions.

Let $\mathbf{S} \in \mathbb{R}_+^{m \times n}$ be a similarity matrix defined via $\mathbf{S}_{ij} = \beta - \mathbf{C}_{ij}$, where the constant $\beta > \max_{ij} \mathbf{C}_{ij}$ ensures non-negativity of $\mathbf{S}$. Then, the following maximization problem is equivalent to (1) as they have the same optimal solution(s):

$$\mathrm{OT}(\boldsymbol{\mu}, \boldsymbol{\nu}) = \max_{\boldsymbol{\gamma} \in \Gamma(\boldsymbol{\mu}, \boldsymbol{\nu})} \langle \mathbf{S}, \boldsymbol{\gamma} \rangle = \beta - \mathrm{OT}_{\min}(\boldsymbol{\mu}, \boldsymbol{\nu}). \tag{2}$$

Partial Optimal Transport (POT) generalizes classical OT by allowing only a subset of the source and/or target mass to be matched (Benamou et al., 2015; Chapel et al.,

2020; Nguyen et al., 2024). A commonly studied variant is the semi-relaxed formulation, suited for unbalanced settings where the target distribution may contain excess mass. Using similarity matrix $\mathbf{S}$, it can be expressed as:

$$\mathrm{POT}(\boldsymbol{\mu},\boldsymbol{\nu}) = \max_{\boldsymbol{\gamma}\in\Gamma_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu})} \langle \mathbf{S},\boldsymbol{\gamma}\rangle \left(= \beta\boldsymbol{\mu}^{\top}\mathbf{1} - \min_{\boldsymbol{\gamma}\in\Gamma_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu})} \langle \mathbf{C},\boldsymbol{\gamma}\rangle \right), \tag{3}$$

where $\Gamma_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu}) = \{\boldsymbol{\gamma} \in \mathbb{R}^{m\times n} \mid \boldsymbol{\gamma} \geq 0, \boldsymbol{\gamma}\mathbf{1} = \boldsymbol{\mu}, \boldsymbol{\gamma}^{\top}\mathbf{1} \leq \boldsymbol{\nu}\}$. It should be noted that $\mathrm{POT}(\boldsymbol{\mu},\boldsymbol{\nu}) = \mathrm{OT}(\boldsymbol{\mu},\boldsymbol{\nu})$ when $\boldsymbol{\mu}^{\top}\mathbf{1} = \boldsymbol{\nu}^{\top}\mathbf{1}$, i.e., the source and the target distributions have equal mass.

Submodularity is a characteristic of set functions that capture diminishing returns: for any sets $A \subseteq B \subseteq V$ and element $u \notin B$, a set function F is submodular if $F(A \cup \{u\}) - F(A) \geq F(B \cup \{u\}) - F(B)$. The term $F(u \mid A) \coloneqq F(A \cup \{u\}) - F(A)$ denotes the *marginal gain* of adding u to A. A function is *monotone* if $F(A) \leq F(B)$ whenever $A \subseteq B$. We provide an alternative definition of submodularity in Section B. For maximizing a non-negative monotone submodular function under a cardinality constraint, i.e., $\max_{S\subseteq V, |S|\leq k} F(S)$, the greedy algorithm achieves a $(1-1/e)$ approximation to the optimal value (Nemhauser et al., 1978).

In the next section 3, we leverage POT to construct a tractable submodular surrogate of the uniform prototype selection problem.

3 PROPOSED APPROACH

Problem setup: Given a source set $\mathcal{S}$ and a target set $\mathcal{T}$, let $\mathcal{P}$ be a candidate prototypical set $\mathcal{P} \subseteq \mathcal{S}$ such that $|\mathcal{P}| \leq k$. The empirical distribution corresponding to set $\mathcal{P}$ may be expressed as $\mu_{\mathcal{P}} = \sum_{i=1}^{m} (\boldsymbol{\mu}_{\mathcal{P}})_i \delta_{\mathbf{x}_i}$, where $\boldsymbol{\mu}_{\mathcal{P}} \in \Delta_m$ and $(\boldsymbol{\mu}_{\mathcal{P}})_i = 0 \ \forall \mathbf{x}_i \notin \mathcal{P}$. Our aim is to find the best prototypical set $\mathcal{P}^*$ such that

- all element of $\mathcal{P}^*$ have equal mass (importance) in the corresponding distribution $\mu_{\mathcal{P}^*} = \sum_{i=1}^{m} (\boldsymbol{\mu}_{\mathcal{P}^*})_i \delta_{\mathbf{x}_i}$, i.e., $(\boldsymbol{\mu}_{\mathcal{P}^*})_i = 1/|\mathcal{P}^*| \ \forall \mathbf{x}_i \in \mathcal{P}^*$, and
- $\boldsymbol{\mu}_{\mathcal{P}^*}$ is *closest* to the underlying target distribution ν under the optimal transport (OT) distance metric.

We note that popular submodular subset selection problems such as facility location or exemplar based clustering (k-medoids) may be viewed as selecting prototypes using the OT metric. In our setup, their optimization objective may be written as

$$\max_{\mathcal{P}\subseteq\mathcal{S}, |\mathcal{P}|\leq k} l(\mathcal{P}), \quad \text{where } l(\mathcal{P}) \coloneqq \max_{\boldsymbol{\gamma}\in\mathbb{R}_{+}^{m\times n}, \boldsymbol{\gamma}^{\top}\mathbf{1}=\boldsymbol{\nu}, \mathrm{supp}(\boldsymbol{\gamma}\mathbf{1})\subseteq\mathcal{P}} \langle \mathbf{S},\boldsymbol{\gamma}\rangle \tag{4}$$

and $\boldsymbol{\nu} \in \Delta_n$ is the given target set distribution, usually set as $\boldsymbol{\nu} = \mathbf{1}/n$. As the objective $l(\mathcal{P})$ may also be written as $l(\mathcal{P}) = \max_{\boldsymbol{\mu}\in\Delta_m, \mathrm{supp}(\boldsymbol{\mu})\subseteq\mathcal{P}} \mathrm{OT}(\boldsymbol{\mu},\boldsymbol{\nu})$, solving (4) implicitly involves learning the underlying distribution of $\mathcal{P}$.

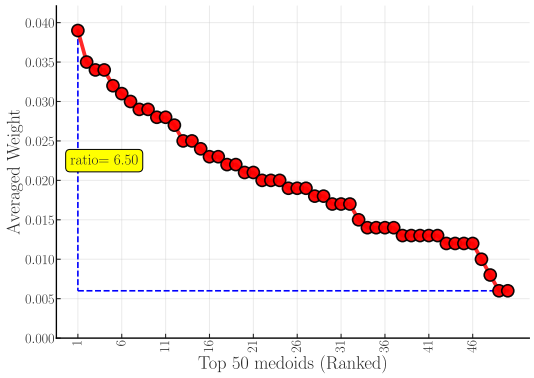


Figure 1: k-medoids (4) consistently learn skewed weights for prototypes on CIFAR10. The plot shows ranked weights of prototypes, averaged over 5 runs.

In particular, if $\hat{\mathcal{P}}$ is a solution of (4) with the corresponding $\boldsymbol{\gamma}_{\hat{\mathcal{P}}}$ such that $l(\hat{\mathcal{P}}) = \langle \mathbf{S}, \boldsymbol{\gamma}_{\hat{\mathcal{P}}} \rangle$, then $\boldsymbol{\mu}_{\hat{\mathcal{P}}} = \boldsymbol{\gamma}_{\hat{\mathcal{P}}}\mathbf{1}$. If the learned $\boldsymbol{\mu}_{\hat{\mathcal{P}}}$ is skewed, it implies that some prototypes have higher mass (importance) than the others. Empirically, this is commonly observed as shown in Figure 1. When one aims to understand the target set $\mathcal{T}$ via the prototypical set $\hat{\mathcal{P}}$ obtained via (4), $\hat{\mathcal{P}}$ and $\boldsymbol{\mu}_{\hat{\mathcal{P}}}$ together provide a representation of $\mathcal{T}$. However, weighted prototypes are hard to interpret, especially for human-in-the-loop scenarios. Skewed prototypical distributions also imply that prototypes receiving low weights contribute less towards the overall objective (4) and may be less influential exemplars. The minority classes often suffer in such cases as they typically receive low weights. Hence, uniformly weighted prototype selection algorithms are desired for fair, unbiased representation and better understanding of datasets.

3.1 Uniform Prototype Selection via Optimal Transport

We propose to alleviate the issue of learning unequally weighted exemplars by enforcing the prototypical distribution to be uniform in the objective. Thus, the proposed uniformly weighted prototype selection problem is as follows:

$$\max_{\mathcal{P}\subseteq\mathcal{S},\ |\mathcal{P}|\leq k} g(\mathcal{P}) \qquad \text{s.t.} \qquad g(\mathcal{P}) = \max_{\boldsymbol{\gamma}\in\Gamma(\mathbf{1}_{\mathcal{P}}/|\mathcal{P}|, \mathbf{1}/n)} \langle \mathbf{S},\boldsymbol{\gamma}\rangle . \tag{5}$$

Here, $g(\mathcal{P})$ denotes the OT objective with uniform marginals $\boldsymbol{\mu}_{\mathcal{P}} = \mathbf{1}_{\mathcal{P}}/|\mathcal{P}|$ and $\boldsymbol{\nu} = \mathbf{1}/n$. Here, $\mathbf{1}_{\mathcal{P}} \in \{0,1\}^m$ represents the set $\mathcal{P}$, i.e., $(\mathbf{1}_{\mathcal{P}})_i = 1$ if $\mathbf{x}_i \in \mathcal{P}$, else 0. It should be noted that in the prototypical distribution $\boldsymbol{\mu}_{\mathcal{P}}$ corresponding to $\mathcal{P}$, all the exemplars $\mathbf{x} \in \mathcal{P}$ are given equal mass in (5). This implies that all the selected exemplars are equally important. We also note that the total mass assigned to the set $\mathcal{S} \setminus \mathcal{P}$ is $1 - \boldsymbol{\mu}_{\mathcal{P}}^{\top}\mathbf{1} = 0$. Empirically, the target set distribution is usually considered uniform $\boldsymbol{\nu} = \mathbf{1}/n$, but our analysis directly extends to non-uniform target set distributions as well. In order to analyze the properties of the

objective in (5), we consider the following proxy problem:

$$\max_{\mathcal{P}\subseteq\mathcal{S},|\mathcal{P}|\leq k} h(\mathcal{P}) \coloneqq |\mathcal{P}|g(\mathcal{P}), \quad \text{where } h(\mathcal{P}) = \mathrm{OT}(\boldsymbol{\mu}_{\mathcal{P}} = \mathbf{1}_{\mathcal{P}}, \boldsymbol{\nu} = |\mathcal{P}|\mathbf{1}/n). \tag{6}$$

We observe that for a given $\mathcal{P}$, if $\boldsymbol{\gamma}_g^*$ is an optimal solution for computing $g(\mathcal{P})$ in (5), then $\boldsymbol{\gamma}_h^* = |\mathcal{P}|\boldsymbol{\gamma}_g^*$ is an optimal solution for computing $h(\mathcal{P})$ in (6) (and vice-versa). Hence, we focus on Problem (6) in the next lemma.

Lemma 1. *The set function $h(\mathcal{P}) : 2^{|\mathcal{S}|} \to \mathbb{R}_+$, defined in (6), satisfies the following properties:*

1. *Non-negativity: $h(\mathcal{P}) \geq 0\ \forall \mathcal{P} \subseteq \mathcal{S}$.*
2. *Monotonicity: $h(\mathcal{P}_2) \geq h(\mathcal{P}_1)\ \forall \mathcal{P}_1 \subseteq \mathcal{P}_2 \subseteq \mathcal{S}$.*
3. *Super-additivity over disjoint sets: $h(\mathcal{P}_1 \cup \mathcal{P}_2) \geq h(\mathcal{P}_1) + h(\mathcal{P}_2)\ \forall \mathcal{P}_1 \cap \mathcal{P}_2 = \phi$.*

The proof of the above result is provided in Appendix B.

Remark 1. Super-additivity is conceptually aligned with increasing returns (supermodularity), just as sub-additivity relates to diminishing returns (submodularity). Hence, maximizing a monotone, non-negative, super-additive function via the greedy algorithm and obtaining approximation guarantees is challenging. To address this, we propose a tight, non-negative, monotone submodular reformulation of Problem (6) in the next section.

3.2 Submodular Reformulation of (6)

We propose the following partial optimal transport (POT) based reformulation of Problem (6):

$$\max_{\mathcal{P}\subseteq\mathcal{S},|\mathcal{P}|\leq k} f(\mathcal{P}), \quad \text{where} \quad f(\mathcal{P}) \coloneqq \mathrm{POT}(\boldsymbol{\mu}_{\mathcal{P}} = \mathbf{1}_{\mathcal{P}}, \boldsymbol{\nu} = k\mathbf{1}/n) = \max_{\boldsymbol{\gamma}\in\Gamma_{\leq}(\mathbf{1}_{\mathcal{P}}, k\mathbf{1}/n)} \langle \mathbf{S}, \boldsymbol{\gamma} \rangle . \tag{7}$$

We note that computing $f(\mathcal{P})$ in (7) is a semi-relaxed optimal transport problem in which the source marginal is tight ($\boldsymbol{\mu}_{\mathcal{P}} = \mathbf{1}_{\mathcal{P}}$) but the target side marginal constraint is relaxed ($\boldsymbol{\nu} \leq k\mathbf{1}/n$). In contrast, computing $h(\mathcal{P})$ in (6) is an OT problem. By relaxing the target side constraint in (7), it is easy to see that $f(\mathcal{P}) \geq h(\mathcal{P}), \forall \mathcal{P}$ with $|\mathcal{P}| \leq k$. We also note that for the sets $\mathcal{P}$ with cardinality k, $f(\mathcal{P}) = \mathrm{OT}(\boldsymbol{\mu}_{\mathcal{P}} = \mathbf{1}_{\mathcal{P}}, \boldsymbol{\nu} = |\mathcal{P}|\mathbf{1}/n) = h(\mathcal{P})$. Our proposed reformulation allows Problem (7) to have certain desirable properties as summarized in our next result.

Lemma 2. *The optimization problem defined in (7) is a non-negative, monotone, submodular maximization problem subject to cardinality constraint k.*

Lemma 2 implies that the classical greedy solution provides the $(1 - 1/e)$ approximation guarantee for (7).

The following lemma proves that Problem (7) is a tight reformulation of Problem (6) and hence we may equivalently solve the relaxed (7) instead of (6), for selecting uniformly weighted prototypes.

Lemma 3. *Let $\mathcal{P}^*$ of cardinality k be an optimal solution of (6). Then $\mathcal{P}^*$ is also an optimal solution of (7), and vice-versa.*

The above analysis ensures that the same approximation guarantee holds for for the super-additive maximization problem (6) as stated below.

Lemma 4. *Let $\hat{\mathcal{P}}$ be the classical greedy solution of (7) with $|\hat{\mathcal{P}}| = k$. Let $\mathrm{OPT} = h(\mathcal{P}^*)$, where $\mathcal{P}^*$ is an optimal solution of (6). Then, $h(\hat{\mathcal{P}}) \geq (1 - 1/e)\mathrm{OPT}$.*

The proof of the above results for Lemmas 2,3,4 are provided in Appendix B.

3.3 Computationally efficient approximate greedy algorithm for (7)

The classical greedy algorithm (Nemhauser et al., 1978) for (7) begins with the empty set $\mathcal{P}_0 = \varnothing$. At iteration $i + 1$, it selects an element $\mathbf{x}^*$ with highest marginal gain, i.e., $\mathbf{x}^* = \arg\max_{\mathbf{x}\in\mathcal{S}\setminus\mathcal{P}_i} f(\mathbf{x}|\mathcal{P}_i)$, and updates $\mathcal{P}_{i+1} = \mathcal{P}_i \cup \{\mathbf{x}^*\}$. For computing the marginal gains of all elements in $\mathcal{S}\setminus\mathcal{P}_i$, $(m-i)$ POT problems (3) need to be solved in the $(i+1)$-th classical greedy iteration. While this number may be reduced using lazy (stochastic) greedy (Minoux, 1978; Mirzasoleiman et al., 2015), the per-iteration cost remains high for large k. Hence, we propose a computationally efficient approximate marginal gain estimator for (7).

In this regard, for a given $\mathcal{P} \subset \mathcal{S}$ such that $|\mathcal{P}| < k$, let $\mathbf{x}_j \in \mathcal{S} \setminus \mathcal{P}$. For notational convenience, let $\mathcal{P}' = \mathcal{P} \cup \{\mathbf{x}_j\}$ and let $\boldsymbol{\gamma}_{\mathcal{P}} = \arg\max_{\boldsymbol{\gamma}\in\Gamma_{\leq}(\mathbf{1}_{\mathcal{P}}, k\mathbf{1}/n)} \langle \mathbf{S}, \boldsymbol{\gamma} \rangle$. We denote by $\hat{\boldsymbol{\gamma}}_{\mathcal{P}'}$ a feasible POT coupling between the sets $\mathcal{P}'$ and $\mathcal{T}$ such that $\hat{\boldsymbol{\gamma}}_{\mathcal{P}'}(\mathcal{I}_{\mathcal{P}}, :) = \boldsymbol{\gamma}_{\mathcal{P}}(\mathcal{I}_{\mathcal{P}}, :)$ and $\hat{\boldsymbol{\gamma}}_{\mathcal{P}'}(j, :) = \mathbf{v}^\top$, where $\mathbf{v} \in \mathbb{R}_+^n$ is a variable. We next construct an estimator of $f(\mathcal{P}')$ as $\hat{f}(\mathcal{P}') = \max_{\hat{\boldsymbol{\gamma}}_{\mathcal{P}'}\in\Gamma_{\leq}(\mathbf{1}_{\mathcal{P}'}, k\mathbf{1}/n)} \langle \mathbf{S}, \hat{\boldsymbol{\gamma}}_{\mathcal{P}'} \rangle$, which is essentially a constrained optimization over $\mathbf{v}$. Our approximate marginal gain function for (7) is as follows:

$$\hat{f}(\mathbf{x}_j \mid \mathcal{P}) = \hat{f}(\mathcal{P}') - f(\mathcal{P}) = \max_{\mathbf{v}\in\mathbb{R}_+^n, \mathbf{v}^\top\mathbf{1}=1, \mathbf{v}\leq\frac{k}{n}\mathbf{1}-\boldsymbol{\gamma}_{\mathcal{P}}^\top\mathbf{1}} \langle \mathbf{S}(j, :), \mathbf{v}^\top \rangle \tag{8}$$

For a given $\boldsymbol{\gamma}_{\mathcal{P}}$, Problem (8) has a closed form expression which involves sorting the vector $\mathbf{S}(j, :)$, i.e., $O(n \log n)$ computation. The following result quantifies the approximation guarantee corresponding to the greedy solution obtained using the proposed approximate marginal gain (8).

Lemma 5. *Let $\alpha_{j,\min}$ denote $\frac{1}{\lfloor n/k \rfloor}$ times the sum of the $\lfloor n/k \rfloor$ smallest entries of the vector $\mathbf{S}(j, :)$, and let $\alpha_{j,\max}$ denote $\frac{1}{\lfloor n/k \rfloor}$ times the sum of the $\lfloor n/k \rfloor$ largest entries of $\mathbf{S}(j, :)$. Define $\alpha = \min_{j\in[m]} \frac{\alpha_{j,\min}}{\alpha_{j,\max}}$. Let $\hat{\mathcal{P}}$ be the solution returned by the greedy algorithm for (7), where the proposed approximate marginal gain function (8) is used in each iteration and $|\hat{\mathcal{P}}| = k$. Then, $f(\hat{\mathcal{P}}) = h(\hat{\mathcal{P}}) \geq (1 - e^{-\alpha})\,\mathrm{OPT}$, where $\mathrm{OPT} = f(\mathcal{P}^*) = h(\mathcal{P}^*)$ and $\mathcal{P}^*$ is an optimal solution to (7) with $|\mathcal{P}^*| = k$.*

Algorithm 1: `UniPROT`

Input: Similarity matrix $\mathbf{S}$ between $\mathcal{S}$ and $\mathcal{T}$, number of prototypes required k, entropic regularization parameter λ
Output: Uniformly weighted prototypical set $\mathcal{P}_k \subseteq \mathcal{S}$ of $\mathcal{T}$

1. $\mathcal{P}_0 \leftarrow \varnothing$
2. for $i = 1$ **to** k **do**
3. $\quad \boldsymbol{\gamma}^*_{\mathcal{P}_i} \leftarrow \arg\max_{\boldsymbol{\gamma} \in \Gamma_{\leq}(\mathbf{1}_{\mathcal{P}_i}, k\mathbf{1}_n/n)} \langle \mathbf{S}, \boldsymbol{\gamma} \rangle - \lambda \langle \boldsymbol{\gamma}, \ln \boldsymbol{\gamma} \rangle$
4. $\quad \mathbf{x}^* \leftarrow \arg\max_{\mathbf{x} \in \mathcal{S} \setminus \mathcal{P}_i} \hat{f}(\mathbf{x} \mid \mathcal{P}_i)$ (Eq. (8))
5. $\quad \mathcal{P}_{i+1} \leftarrow \mathcal{P}_i \cup \{\mathbf{x}^*\}$
6. end for
7. return $\mathcal{P}_k$

- **Steps 2–5** are iteratively executed until the cardinality constraint k is satisfied.
- **Step 3** At each iteration, it updates the transport plan $\boldsymbol{\gamma}^*_{\mathcal{P}_i}$ and selects the element $\mathbf{x}^*$ that maximizes the approximate marginal gain defined in Eq. (8).
- **Step 4** selects the element $\mathbf{x}^*$ that maximizes the approximate marginal gain $\hat{f}(\cdot)$ across the candidate search space $\mathcal{S} \setminus \mathcal{P}_i$. (*Note*: For larger search space i.e. where $|\mathcal{S}|$ is large enough, we consider a *Stochastic-Greedy* version instead of a Naive Greedy approach where the candidate search space is considered as $\mathcal{R} \subseteq \mathcal{S} \setminus \mathcal{P}_i$ with $nk^{-1}\log(1/\epsilon)$ elements which are selected uniformly at random.

The proof of the above result is provided in Appendix B. The key idea in the proof methodology is to lower bound the proposed approximate marginal gain (8) as a fraction of the true marginal gain $f(\mathbf{x}_j \mid \mathcal{P})$.

We observe that the proposed approximate marginal gain-based greedy algorithm yields theoretical guarantees for (6) that are equivalent to those established for the maximization of an α-weakly submodular function (Das and Kempe, 2018a; Elenberg et al., 2018).

4 ALGORITHM DETAILS

Here, we present our algorithm for `UniPROT` (Alg. 1). At each iteration, we first compute a partial optimal transport plan for the current set and then use it to evaluate the approximate marginal gain $\hat{f}(\cdot)$ over the remaining candidates, selecting the best next prototype. To solve the partial optimal transport subproblem in Step 3, we use Bregman iterations (Benamou et al., 2015).

Computation Cost. Overall, finding the (approximate) next best element $\mathbf{x}^*$ requires solving a *single* POT problem of dimension $(i+1) \times n$ along with $O((m-i)n \log n)$ additional computations. The POT problem can be efficiently solved in $O(i \cdot n)$ using the Bregman-Dykstra iterations (Benamou et al., 2015) or the Sinkhorn algorithm (Cuturi, 2013; Chapel et al., 2020) by adding a small entropic regularization in (3). Hence, the computational cost of the proposed `UniPROT` algorithm for selecting k uniformly weighted prototypes is $O(kmn \log n)$. This cost can be reduced to $O(kmn)$ by utilizing an additional $m \times n$ memory to store the sorted rows of $\mathbf{S}$, which is a one-time preprocessing step of $O(mn \log n)$ computations. Consequently, our algorithm selects equally important prototypes with an overall computational cost that closely matches that of the classical greedy algorithm for solving (4). We term our approach `UniPROT`.

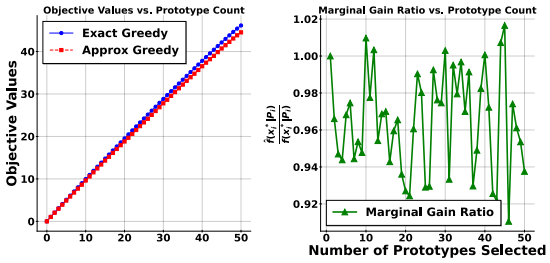


Figure 2: Exact marginal gain versus the proposed approximate marginal gain (8) on MNIST.

Comparing Approximate vs Exact Marginal gain: We evaluate the effectiveness of the proposed computationally efficient approximate marginal gain (8). For this, we compare the objective value $f(\mathcal{P})$ under the two next element selection settings: (a) exact marginal gain, $\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathcal{S} \setminus \mathcal{P}} f(\mathbf{x}|\mathcal{P})$, and (b) approximate marginal gain (8), $\mathbf{x}^* = \arg\max_{\mathbf{x} \in \mathcal{S} \setminus \mathcal{P}} \hat{f}(\mathbf{x}|\mathcal{P})$.

In Figure 2, we plot the objective values and the ratio of approximate marginal gain and exact marginal gain across greedy iterations. We observe that, across iterations, the two objectives are very close and the ratio $\hat{f}(\mathbf{x}^*_i|\mathcal{P}_i)/f(\mathbf{x}^*_i|\mathcal{P}_i)$ is close to its maximum value 1. This implies that the proposed approximate marginal gain (8) of (7) is a good computationally efficient alternative to exact marginal gain. We provide additional results on how different choices of entropic regularization as part of our implementation affects this approximation gap in Appendix E.1.

5 EXPERIMENTAL EVALUATION

We now empirically validate the utility of `UniPROT` based uniform weighted prototype selection in various domains.

5.1 Long Tailed Image Classification

We assess the effectiveness of the representative samples selected by the weighted prototype selection method (4) and our proposed uniformly weighted variant, UniPROT, by evaluating the performance of the corresponding nearest prototype classifiers (Bien and Tibshirani, 2011; Kim et al., 2016; Gurumoorthy et al., 2021) in imbalanced multiclass classification setting.

Experimental Setup. Let $\mathcal{S}$ and $\mathcal{T}$ denote source and target datasets, respectively, such that $\mathcal{S} \cap \mathcal{T} = \varnothing$. Source set $\mathcal{S}$ has same number of samples from all classes while the target set $\mathcal{T}$ exhibit a skewed class distribution. A skewed target set distribution simulate real-world scenarios involving non-trivial marginal shifts. The label information is not available during the prototype selection phase, *i.e.*, prototype selection is completely unsupervised. Let $\mathcal{P} \subseteq \mathcal{S}$ be a candidate prototypical set intended to model the target dataset T. After the set $\mathcal{P}$ is obtained, class labels of prototypical examples are now made available. We next parameterize a 1-nearest neighbor (1-NN) classifier with the prototypes in $\mathcal{P}$ (see Appendix D.5) and using it to classify the data points in $\mathcal{T}$. Overall, the performance of the 1-NN classifier parameterized with $\mathcal{P}$ is an indicator of how representative $\mathcal{P}$ is of the target $\mathcal{T}$ (Bien and Tibshirani, 2011; Gurumoorthy et al., 2021).

Datasets. We consider the MNIST dataset and long-tailed versions of CIFAR-LT (Krizhevsky et al., 2009) datasets. The latter was obtained using the long-tail experimental setup of (Menon et al., 2021). For MNIST, we obtain a skewed target set distribution by ensuring that two (randomly) chosen class constitute $k\%$ (each) of $|\mathcal{T}|$ and the remaining $(100-2k)\%$ is spread uniformly over the other classes. Additional datasets are provided in Appendix E.

Results. In Figure 3, we observe that the proposed UniPROT improves the minority class performance over k-medoids (4) which selects weighted prototypes (Gurumoorthy et al., 2021). It should be noted that the learned weights of the latter were not employed during the inference stage as it deteriorates the performance.

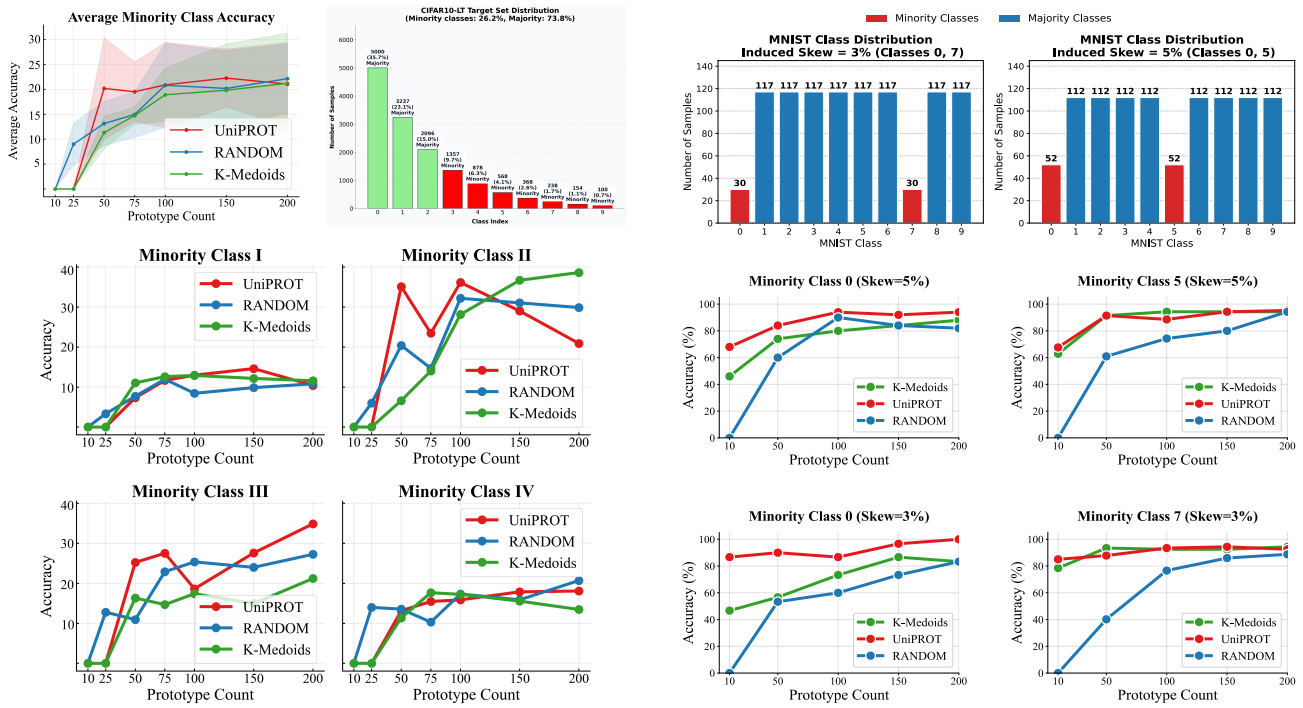
5.2 High quality Mini-Batch Selection for LLM training

Training LLMs with large mini-batches is known to accelerate convergence and improve model performance. However, this approach is often impractical due to the substantial memory overheads. A common workaround is to select representative samples from a mini-batch that approximate the gradient of larger batches (Mirzasoleiman et al., 2020a; Yang et al., 2023). Existing work (Killamsetty et al., 2021a,b; Wang et al., 2024; Nguyen et al., 2025) rely on facility location or k-medoids based subset selection (4) to identify small high-quality mini-batches for LLM training. As discussed previously, these subsets approaches implicitly learn weighted representation. As the LLM training data is usually a highly imbalanced mixture of sources, weighted subset selection (4) may choose more representative prototypes with higher weights for larger sources and low-quality prototypes with low weights for smaller sources. However, this leads to misrepresentation of smaller sources and an eventual suboptimal performance of the training LLMs. To overcome this difficultly, we employ our uniformly weighted subset selection approach, UniPROT, in this problem setting and illustrate its suitability.

Problem Setting. Consider a dataset $\mathcal{V} = \{\mathcal{V}_1 \cup \cdots \cup \mathcal{V}_p \cup \mathcal{V}_{p+1} \cup \cdots \cup \mathcal{V}_Q\}$, with Q sources, where the first p sources correspond to minority sources and the remaining are majority sources. At iteration t, we have a (random) batch $\mathcal{B}_t$ which would typically have instances from all the sources. The aim is to select a highly representative subset of $\mathcal{B}_t$. One can perform this subset selection source-wise (Nguyen et al., 2025), i.e., independently select k_q instances from $\mathcal{B}_q^t = \mathcal{B}_t \cap \mathcal{V}_q \ \forall q$ where the budget k_q is such that $\sum_q k_q = k$. Alternatively, one can directly select k prototypes from $\mathcal{B}_t$.

Gradient based Representation for Subset Selection. In order to select a subset of batch $\mathcal{B}_t$, we require a feature representation of the data points which is relevant to the subset selection problem. Recent works (Mirzasoleiman et al., 2020b; Killamsetty et al., 2021a; Wang et al., 2024) have demonstrated the utility of the gradients as feature representation of the data points. Hence, during model training at iteration t, the similarity (or cost) matrix may be computed using the gradients of the data points in $\mathcal{B}_t$ (and also validation data points in case of (Wang et al., 2024)). However, as the dimensionality of gradients in LLMs is very large, employing exact gradients for subset selection problem may become impractical especially with high mini-batch size or low-resource hardware. Hence, (Nguyen et al., 2025) employed computationally efficient zeroth-order gradient approximation methods for constructing the similarity matrix for the subset selection problem. Overall, let the (gradient-based) representation of data points $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{B}_t$ be $\mathbf{g}^t(\mathbf{x}_i)$ and $\mathbf{g}^t(\mathbf{x}_j)$, respectively, in the t-th iteration. Then, we employ the similarity matrix $\mathbf{S}(i,j) = \left\langle \mathbf{g}^t(\mathbf{x}_i), \mathbf{g}^t(\mathbf{x}_j) \right\rangle$ in (7) for our method UniPROT. On the other hand, (Nguyen et al., 2025) observed better results with ℓ_1 distance based similarity matrix, i.e., $\mathbf{S}(i,j) = c - \|\mathbf{g}^t(\mathbf{x}_i) - \mathbf{g}^t(\mathbf{x}_j)\|_1$, where c is a large constant which ensures the similarity matrix has positive entries. We also note that as (Wang et al., 2024) requires a validation dataset U_t, it computes both $\left\langle \mathbf{g}^t(\mathbf{x}_i), \mathbf{g}^t(\mathbf{x}_j) \right\rangle \ \forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{B}_t$ and $\left\langle \mathbf{g}^t(\mathbf{x}_i), \mathbf{g}^t(\mathbf{x}_{\texttt{val}}) \right\rangle \ \forall \mathbf{x}_i \in \mathcal{B}_t, \mathbf{x}_{\texttt{val}} \in U_t$.

UniPROT-PS (Per Source). Given each batch consists of samples drawn from multiple sources, the objective here is to perform prototype selection at a per-source level. In particular, (Nguyen et al., 2025) ensures that samples belonging to minority sources are also preserved in the final selection, while prototype selection is done only for major-



(a) CIFAR10-LT (b) MNIST (Synthetic-LT)

Figure 3: *Minority Class Accuracy Analysis*:On CIFAR10-LT and synthetic MNIST, `UniPROT` outperforms k-medoids on minority classes. (**Top Left**): Average minority-class accuracy vs. prototype count. (**Bottom Left**): CIFAR10 minority class-wise accuracy (avg. over 3 runs). (**Top Right**): MNIST with Induced Skew: (i) classes 0 and 5 both at 5% each (other classes at 90%), and (ii) classes 0 and 7 both at 3% each (other classes at 94%). (**Bottom Right**): `UniPROT` on average out-performs k-medoids across different skew variations for minority classes.

Table 1: Performance across in-domain and out-of-domain datasets for **PHI-3** on **MATHINSTRUCT**, batch size $|\mathcal{B}| = 128$ and total budget $k = 64$. For completeness and a **fair evaluation**, results are reported under two configurations for **all** baselines: *source-wise*(left of "/") and *batch-wise* (right of "/"). `UniPROT` consistently beats baselines in both configurations.

Method	In-domain				Out-of-domain				Avg-All
	GSM8K	MATH	NumGLUE	Avg	SVAMP	Mathematics	SimulEq	Avg	
FT	76.72	36.54	62.57	58.61	85.10	33.30	62.78	60.39	59.50
MaxLoss	70.64 / 69.44	32.05 / 30.02	57.80 / 58.90	53.50 / 52.79	80.60 / 79.20	31.45 / 30.06	57.19 / 55.82	56.41 / 55.03	54.96 / 53.91
GradNorm	76.04 / 75.40	36.10 / 35.03	64.01 / 64.10	58.72 / 58.18	85.30 / 84.17	38.00 / 36.50	61.84 / 65.70	61.71 / 62.12	60.22 / 60.15
SBERT	73.20 / 72.80	35.54 / 35.06	60.26 / 57.60	56.33 / 55.15	79.31 / 77.90	34.05 / 34.00	61.70 / 58.90	58.35 / 56.93	57.34 / 56.04
COLM	76.80 / 76.36	37.28 / 36.42	64.11 / 64.10	59.40 / 58.96	85.10 / 85.30	**38.00** / 37.40	62.25 / 63.60	61.78 / 62.10	60.59 / 60.53
GREATS	76.72 / 77.80	37.84 / 37.28	67.46 / 64.40	60.67 / 59.83	86.10 / 85.00	35.60 / **38.19**	62.06 / 61.92	61.25 / 61.64	60.96 / 60.73
`UniPROT` (Ours)	**79.07 / 78.16**	**38.40 / 37.76**	**68.80 / 66.02**	**62.09 / 60.65**	**86.20 / 85.70**	36.90 / 37.20	**66.73 / 68.28**	**63.28 / 63.73**	**62.68 / 62.19**

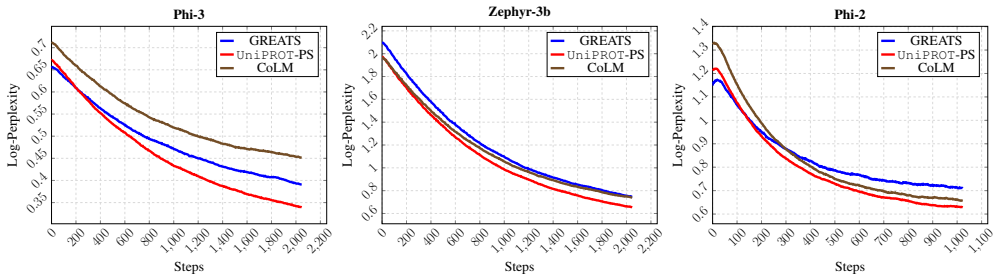


Figure 4: Validation perplexity dynamics on PHI-3, ZEPHYR-3B and PHI-2 during training vs top 3 best-performing baselines on **MATHINSTRUCT**. `UniPROT`-PS consistently outperforms other baselines.

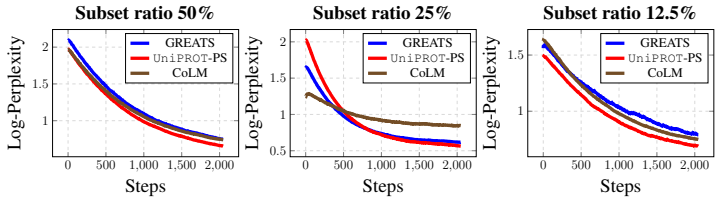


Figure 5: Validation log-pplx with changing prototype percentage (left) and ablation table (right) on ZEPHYR-3B.

Table 2: Variation with λ (entropic regularization).

λ (Entropic Reg.)	GSM-8k	NumGlue	Svamp
0.1	47.76	37.5	52.9
0.05	48.36	36.1	54
0.01	49.40	36.7	54.5

ity sources, thus preventing less representation of minority sources in the final selected subset. We consider UniPROT at each source level per batch with each source having some cardinality constraint. The POT problem can then be formulated as $f(\mathcal{P}_q) \coloneqq \max_{\boldsymbol{\gamma} \in \Gamma_{\leq}(\mathbf{1}_{\mathcal{P}_q}, k_q \mathbf{1}/n)} \langle \mathbf{S}_q, \boldsymbol{\gamma} \rangle$ where $\mathcal{P}_q$ indicates the prototypes per q-th data source to be selected and k_q indicates the cardinality per source and n_q indicates the total size of the samples per source in the batch $\mathcal{B}_t$ (i.e., $n_q = |\mathcal{B}_q^t|$). Hence, the selected subset would be $\bigcup_{q=1} \mathcal{P}_q$. Here, we define $\mathbf{S} \in \mathbb{R}^{|\mathcal{B}_q^t| \times |\mathcal{B}_q^t|}$ where both the source and target are same $\mathcal{S} = \mathcal{T} = \mathcal{B}_q^t$ (qth data source in the batch).

UniPROT-PB (Per Batch). Beyond per-source prototype selection, we also consider the joint selection of prototypes across the entire batch, i.e., over all samples aggregated from multiple sources. This broader perspective is particularly beneficial, as UniPROT mitigates the risk of systematically down-weighting minority sources, a bias that methods such as k-medoids or Facility Location clustering may inadvertently introduce as we observe in Figure 3. Here, the similarity matrix $\mathbf{S} \in \mathbb{R}^{|\mathcal{B}_t| \times |\mathcal{B}_t|}$ is formed in all samples within the entire batch.

Models and Training details. We evaluate on PHI-2 (2.7B) (Javaheripi et al., 2023), PHI-3(3.8B) (Li et al., 2023), and ZEPHYR (3B) (Tunstall et al., 2023). For finetuning, we employ LoRA with rank 128, α = 512, and dropout 0.05. Following (Nguyen et al., 2025), the LoRA are applied to all attention matrices (QKV PROJ) and two fully connected layers for the PHI models; for ZEPHYR, adapters are applied to all attention matrices (QKVO PROJ). All experiments are conducted on 3×A6000 GPUs.

Baselines. We compare UniPROT against (i) standard finetuning (FT), (ii) recent mini-batch selection approaches such as GREATS (Wang et al., 2024) and COLM (Nguyen et al., 2025), and (iii) one-shot selection strategies such as Grad Norm (GN) (Katharopoulos and Fleuret, 2018) and MaxLoss (Shalev-Shwartz and Wexler, 2016), adapted to mini-batch selection setting. For completeness and a **fair evaluation**, we include baseline results under both source-wise (**PS**) and batch-wise (**PB**) settings, whenever relevant. These are defined analogously to UniPROT-PS and UniPROT-PB. For the Full-Finetuning (**FT**) baseline, the model is trained as usual without any prototype selection. We note that (Nguyen et al., 2025) employs k-medoids for subset selection and demonstrate source-wise selection being better than batch-wise selection. On the other hand, (Wang et al., 2024) assumes access to a validation set and their subset selection criterion could be viewed as optimizing maximum mean discrepancy between the prototypical set (a subset of $\mathcal{B}_t$) and the validation set (with gradient based features and linear kernels). For a fair comparison, we simply assume random subset of training set as "validation anchors" for (Wang et al., 2024).

Finetuning Datasets. We train on the MATHINSTRUCT dataset (Yue et al., 2023), which consists of 260K instruction tuning samples curated from 14 highly imbalanced data sources. In addition, on the SUPERGLUE benchmark (Wang et al., 2019) for the following classification tasks *SST-2*, *CB*, and *MultiRC*. We note that SUPERGLUE does not have source information.

Evaluation datasets. Following (Yue et al., 2023), we evaluate the finetuned models on both in-domain and out-of-domain benchmarks. The in-domain set comprises GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), and NumGLUE (Mishra et al., 2022), whereas the out-of-domain set includes SVAMP (Patel et al., 2021), Mathematics (Davies et al., 2021), and SimulEq (Koncel-Kedziorski et al., 2016).

5.3 Evaluation Results and Discussion

Finetuning Experiments: We train all models for 2048 steps with batch size of 128 and prototype-ratio as 50% (thus effective batch size k=64), except for Full-Finetuning (**FT**). The Full-Finetuning baselines train on the whole input-batch as is standard. Table 3 presents the results on batch selection where we train all baselines using batch selection for a fair comparison and UniPROT performs well in full-batch selection. Table 9 shows the results of source-wise finetuning on MATHINSTRUCT. We do source selection on **all** baselines for fair comparision. The table indicates that UniPROT is significantly better than other baselines in both the in-domain and out-of-domain settings.

Source unavailable datasets: In some benchmarks, such as SUPERGLUE, source annotations are not provided, making source-wise selection infeasible. This setting is common in practice, where fine-tuning data may arrive without clear domain tags. To test robustness, we evaluate on SST2, MultiRC, and CB under this

Table 3: Comparison of performance of batch-wise selection across baselines for **PHI-3** in **SUPERGLUE** (Wang et al., 2019).

Method	SST2	MultiRC	CB	Avg
FT	93.91	86.05	92.72	90.89
GradNorm	87.94	57.54	69.10	71.53
SBERT	90.10	82.11	87.27	86.49
CoLM	94.72	82.99	93.05	90.25
GREATS	**94.81**	**88.42**	93.12	92.12
UniPROT-PB (Ours)	94.65	88.03	**94.54**	**92.41**

source-agnostic regime. Since ground-truth sources are unavailable, UniPROT-PS is omitted and all baselines are compared via batch selection for fairness. We train for 512 steps, batch-size 32 and selection ratio of 25%, except for FT where we train on whole batch. As shown in Table 3, UniPROT still outperforms alternatives, indicating strong performance without source information.

Pretraining Experiments To test the effectiveness of UniPROT, we conduct pretraining experiments with OpenWebText on LLaMA-3 500M and 60M models for 20k steps. All baselines are trained in *batch-wise*(PB) mode. We defer the details to Appendix C.6. Figure 6 indicates that UniPROT outperforms all baselines including Full-batch pretraining both at large and small scale models.

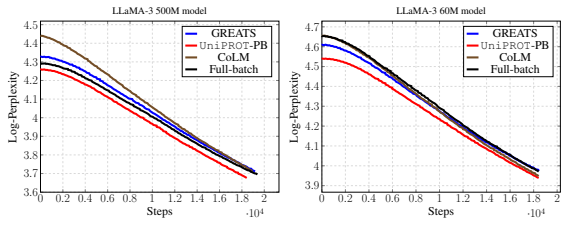


Figure 6: Perplexity dynamics on **Pretraining** LLAMA-3 500M and 60M for 20k steps.

Effect of number of selected prototypes. We finetune ZEPYR-3B on MATHINSTRUCT for 2048 steps while varying the selection ratio $r \in 50\%, 25\%, 12.5\%$. Figure 5 reports the validation perplexity. We observe that UniPROT remains consistently stable across all ratios. In contrast, COLM degrades noticeably as r decreases, while GREATS shows a smaller but still measurable rise. Overall, UniPROT exhibits the lowest sensitivity to prototype budget, indicating stronger robustness.

Effect of regularization on downstream performance. We ablate the entropic regularization coefficient in the partial optimal transport objective by finetuning PHI-3 on MATHINSTRUCT and evaluating downstream on GSM8K, NumGLUE and Svamp. We finetune for 128 steps and 20 OT iterations.(Table 2). With $\lambda = 0.1$, performance is consistently lower, while reducing to $\lambda = 0.01$ yields clear gains on both tasks. This trend aligns with prior observations (Cuturi, 2013), where smaller regularization improves transport fidelity and leads to better downstream accuracy.

6 CONCLUSION

We proposed UniPROT, a scalable and theoretically grounded framework for selecting k uniformly weighted prototypes that summarize a target distribution via optimal transport. This leads to a super-additive maximization problem under cardinality constraints, for which we introduced a novel submodular reformulation with a provably tight equivalence at k, enabling a greedy algorithm with a $(1-1/e)$ approximation guarantee. UniPROT consistently improves minority-class representation in imbalanced classification tasks and enhances mini-batch quality for training large language models, outperforming existing methods in both accuracy and efficiency. By imposing uniform weights, UniPROT mitigates bias toward majority classes, and supporting equitable learning.

7 ACKNOWLEDGEMENTS

PC acknowledges the Microsoft Research India PhD Award and Prime Minister Research Fellowship to support this research work. GR thanks Bank of Baroda Chair Professorship. PJ acknowledges the support of Anusandhan National Research Foundation under ARG-MATRICS program and IIT Bombay seed grant.

References

Adam, K. D. B. J. et al. (2014). A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 1412(6).

Agueh, M. and Carlier, G. (2011). Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924.

Benamou, J.-D., Carlier, G., Cuturi, M., Nenna, L., and Peyré, G. (2015). Iterative bregman projections for regularized transportation problems. *SIAM Journal on Scientific Computing*, 37(2):A1111–A1138.

Bien, J. and Tibshirani, R. (2011). Prototype selection for interpretable classification.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. (2018). JAX: composable transformations of Python+NumPy programs.

Chapel, L., Alaya, M. Z., and Gasso, G. (2020). Partial optimal tranport with applications on positive-unlabeled learning. In *NeurIPS*.

Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., and Su, J. K. (2019). This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano,

R., et al. (2021). Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Cuturi, M. (2013). Lightspeed computation of optimal transportation distances. *Advances in Neural Information Processing Systems*, 26(2):2292–2300.

Cuturi, M. and Doucet, A. (2014). Fast computation of wasserstein barycenters. In *International conference on machine learning*, pages 685–693. PMLR.

Das, A. and Kempe, D. (2018a). Approximate submodularity and its applications: Subset selection, sparse approximation and dictionary selection. *Journal of Machine Learning Research*, 19(3):1–34.

Das, A. and Kempe, D. (2018b). Approximate submodularity and its applications: Subset selection, sparse approximation and dictionary selection. *Journal of Machine Learning Research*, 19(3):1–34.

Davies, A., Veličković, P., Buesing, L., Blackwell, S., Zheng, D., Tomašev, N., Tanburn, R., Battaglia, P., Blundell, C., Juhász, A., Lackenby, M., Williamson, G., Hassabis, D., and Kohli, P. (2021). Advancing mathematics by guiding human intuition with AI. *Nature*, 600(7887):70–74.

Dhurandhar, A. and Gurumoorthy, K. S. (2020). Classifier invariant approach to learn from positive-unlabeled data. In *IEEE ICDM*.

Elenberg, E. R., Khanna, R., Dimakis, A. G., and Negahban, S. (2018). Restricted strong convexity implies weak submodularity. *Annals of Statistics*, 46(6B):3539–3568.

Gurumoorthy, K. S., Dhurandhar, A., Cecchi, G., and Aggarwal, C. (2019). Efficient data representation by selecting prototypes with importance weights. In *IEEE ICDM*.

Gurumoorthy, K. S., Jawanpuria, P., and Mishra, B. (2021). Spot: A framework for selection of prototypes using optimal transport. In *ECML PKDD*.

Hendrycks, D., Burns, C., Kadavath, S., Arora, A., Basart, S., Tang, E., Song, D., and Steinhardt, J. (2021). Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Ho, N., Nguyen, X., Yurochkin, M., Bui, H. H., Huynh, V., and Phung, D. (2017). Multilevel clustering via wasserstein means. In *International conference on machine learning*, pages 1501–1509. PMLR.

Hong, F., Lyu, Y., Yao, J., Zhang, Y., Tsang, I., and Wang, Y. (2024). Diversified batch selection for training acceleration. In *International Conference on Machine Learning*, pages 18648–18667. PMLR.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., et al. (2022). Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3.

Javaheripi, M., Bubeck, S., Abdin, M., Aneja, J., Bubeck, S., Mendes, C. C. T., Chen, W., Del Giorno, A., Eldan, R., Gopi, S., et al. (2023). Phi-2: The surprising power of small language models. *Microsoft Research Blog*.

Kantorovich, L. (1942). On the transfer of masses (in russian). *Doklady Akademii Nauk*, 37(2):227–229.

Katharopoulos, A. and Fleuret, F. (2018). Not all samples are created equal: Deep learning with importance sampling. In *International conference on machine learning*, pages 2525–2534. PMLR.

Kawano, K., Koide, S., and Otaki, K. (2022). Partial Wasserstein covering. In *AAAI*.

Killamsetty, K., Durga, S., Ramakrishnan, G., De, A., and Iyer, R. (2021a). Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pages 5464–5474. PMLR.

Killamsetty, K., Sivasubramanian, D., Ramakrishnan, G., and Iyer, R. (2021b). 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.

Kim, B., Khanna, R., and Koyejo, O. O. (2016). Examples are not enough, learn to criticize! criticism for interpretability. In *NeurIPS*.

Koncel-Kedziorski, R., Roy, S., Amini, A., Kushman, N., and Hajishirzi, H. (2016). Mawps: A math word problem repository. In *Proceedings of the 2016 conference of the north american chapter of the association for computational linguistics: human language technologies*, pages 1152–1157.

Kothawade, S., Kaushal, V., Ramakrishnan, G., Bilmes, J. A., and Iyer, R. K. (2021). Submodular mutual information for targeted data subset selection. *CoRR*, abs/2105.00043.

Krause, A. and Golovin, D. (2014). Submodular function maximization. *Tractability*, 3(71-104):3.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Li, Y., Bubeck, S., Eldan, R., Del Giorno, A., Gunasekar, S., and Lee, Y. T. (2023). Textbooks are all you need ii: phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Lin, H. and Bilmes, J. (2011). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*.

Liu, H., Li, Y., Xing, T., Wang, P., Dalal, V., Li, L., He, J., and Wang, H. (2025). Dataset distillation via the wasserstein metric. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1205–1215.

Liu, Z., Karbasi, A., and Rekatsinas, T. (2024). Tsds: Data selection for task-specific model finetuning. *Advances in Neural Information Processing Systems*, 37:10117–10147.

Menon, A. K., Rawat, A. S., Reddi, S., Kim, S., and Kumar, S. (2021). A statistical perspective on distillation. In *International Conference on Machine Learning*, pages 7632–7642. PMLR.

Minoux, M. (1978). Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*.

Mirzasoleiman, B., Badanidiyuru, A., Karbasi, A., Vondrák, J., and Krause, A. (2015). Lazier than lazy greedy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29.

Mirzasoleiman, B., Bilmes, J., and Leskovec, J. (2020a). Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*.

Mirzasoleiman, B., Cao, K., and Leskovec, J. (2020b). Coresets for robust training of deep neural networks against noisy labels. *Advances in Neural Information Processing Systems*, 33:11465–11477.

Mishra, S., Mitra, A., Varshney, N., Sachdeva, B., Clark, P., Baral, C., and Kalyan, A. (2022). Numglue: A suite of fundamental yet challenging mathematical reasoning tasks. *arXiv preprint arXiv:2204.05660*.

Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14:265–294.

Nguyen, A. D., Nguyen, T. D., Nguyen, Q. M., Nguyen, H. H., Nguyen, L. M., and Toh, K.-C. (2024). On partial optimal transport: Revising the infeasibility of sinkhorn and efficient gradient methods. In *AAAI*.

Nguyen, D., Yang, W., Anand, R., Yang, Y., and Mirzasoleiman, B. (2025). Mini-batch coresets for memory-efficient language model training on data mixtures. In *International Conference on Learning Representations*.

Nguyen, T., Novak, R., Xiao, L., and Lee, J. (2021). Dataset distillation with infinitely wide convolutional networks. *Advances in Neural Information Processing Systems*, 34:5186–5198.

Patel, A., Bhattamishra, S., and Goyal, N. (2021). Are nlp models really able to solve simple math word problems? *arXiv preprint arXiv:2103.07191*.

Peyré, G., Cuturi, M., et al. (2019). Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Riaz, B., Karahan, Y., and Brockmeier, A. J. (2023). Partial optimal transport for support subset selection. *Transactions on Machine Learning Research*.

Schlegel, M., Pan, Y., Chen, J., and White, M. (2017). Adapting kernel representations online using submodular maximization. In *Proceedings of the 34th International Conference on Machine Learning*.

Shalev-Shwartz, S. and Wexler, Y. (2016). Minimizing the maximal loss: How and why. In *International Conference on Machine Learning*, pages 793–801. PMLR.

Solso, R. L., MacLin, O. H., and MacLin, M. K. (2017). *Cognitive Psychology*. Pearson Education.

Tan, H., Wu, S., Huang, W., Zhao, S., and QI, X. (2025). Data pruning by information maximization. In *The Thirteenth International Conference on Learning Representations*.

Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., Von Werra, L., Fourrier, C., Habib, N., et al. (2023). Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*.

Wang, A., Pruksachatkun, Y., Nangia, N., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2019). Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Wang, J., Dai, T., Zhang, B., Yu, S., Lim, E. G., and Xiao, J. (2025). Pot: Prototypical optimal transport for weakly supervised semantic segmentation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15055–15064.

Wang, J. T., Wu, T., Song, D., Mittal, P., and Jia, R. (2024). Greats: Online selection of high-quality data for llm training in every iteration. *Advances in Neural Information Processing Systems*, 37:131197–131223.

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

Yue, X., Qu, X., Zhang, G., Fu, Y., Huang, W., Sun, H., Su, Y., and Chen, W. (2023). Mammoth: Building math generalist models through hybrid instruction tuning. *arXiv preprint arXiv:2309.05653*.

Zhang, G., Zhang, H., Wang, Y., Li, R., Tan, H., and Liang, J. (2024). Hyperspherical multi-prototype with optimal transport for event argument extraction. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9271–9284.

Zhao, B., Mopuri, K. R., and Bilen, H. (2021). Dataset condensation with gradient matching. In *ICLR*.

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

Checklist

The checklist follows the references. For each question, choose your answer from the three possible options: Yes, No, Not Applicable. You are encouraged to include a justification to your answer, either by referencing the appropriate section of your paper or providing a brief inline description (1-2 sentences). Please do not modify the questions. Note that the Checklist section does not count towards the page limit. Not including the checklist in the first submission won't result in desk rejection, although in such case we will ask you to upload it during the author response period and include it in camera ready (if accepted).

In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]
2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Yes]
   (c) Clear explanations of any assumptions. [Yes]
3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:

(a) Citations of the creator If your work uses existing assets. [Yes]

(b) The license information of the assets, if applicable. [Not Applicable]

(c) New assets either in the supplemental material or as a URL, if applicable. [Not Applicable]

(d) Information about consent from data providers/curators. [Not Applicable]

(e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:

(a) The full text of instructions given to participants and screenshots. [Not Applicable]

(b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]

(c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Supplementary Material: Uniform Prototype Selection via Partial Optimal Transport with Submodular Guarantees

Contents

Supplementary Material: Uniform Prototype Selection via Partial Optimal Transport with Submodular Guarantees

A ORGANIZATION OF APPENDIX

The Appendix is structured as follows. Section B summarizes the main theoretical results. Section 4 presents the detailed description of the proposed algorithm. Sections C and D outline the implementation aspects and the specifics of the experimental setup, respectively. Finally, Section G discusses the broader impact of our work, and Section H provides a link to the publicly available codebase.

B THEORETICAL RESULTS

Submodularity Ratio The notion of submodularity ratio is given by approximate submodularity in (Das and Kempe, 2018b). For a monotone function f the submodularity ratio w.r.t a set S and a parameter $k \geq 0$ as

$$\alpha_{L,K}(f) = \min_{\substack{\mathcal{S}\subseteq L,\, \mathcal{A}\subseteq L \\ |\mathcal{A}|\leq K,\, \mathcal{A}\cap\mathcal{S}=\emptyset}} \frac{\sum_{u\in A} f(\mathcal{S}\cup\{u\}) - f(\mathcal{S})}{f(\mathcal{S}\cup A) - f(\mathcal{S})}, \quad \text{with } \frac{0}{0} := 1.$$

f is submodular if and only if $\alpha_{L,K}(f) \geq 1$. If the ratio

$$\alpha := \frac{\sum_{u\in A} f(\mathcal{S}\cup\{u\}) - f(\mathcal{S})}{f(\mathcal{S}\cup\mathcal{A}) - f(\mathcal{S})}$$

is strictly positive but not necessarily greater than 1, then f is said to be α-weakly submodular.

Lemma 1. *The set function $h(\mathcal{P}) : 2^{|\mathcal{S}|} \to \mathbb{R}_+$, defined in (6), satisfies the following properties:*

1. *Non-negativity: $h(\mathcal{P}) \geq 0\ \forall \mathcal{P} \subseteq \mathcal{S}$.*
2. *Monotonicity: $h(\mathcal{P}_2) \geq h(\mathcal{P}_1)\ \forall \mathcal{P}_1 \subseteq \mathcal{P}_2 \subseteq \mathcal{S}$.*
3. *Super-additivity over disjoint sets: $h(\mathcal{P}_1 \cup \mathcal{P}_2) \geq h(\mathcal{P}_1) + h(\mathcal{P}_2)\ \forall \mathcal{P}_1 \cap \mathcal{P}_2 = \phi$.*

Proof. We prove each property in turn (refer to the definition of $h(\mathcal{P})$ in (6)).

1. Non-negativity. The non-negativity follows from the definition of $h(\cdot)$ in (6) namely, $h(\mathcal{P}) := \max_{\boldsymbol{\gamma}\in\Gamma(\mathbf{1}_{\mathcal{P}}, |\mathcal{P}|\mathbf{1}/n)} \langle \mathbf{S}, \boldsymbol{\gamma}\rangle$, where the similarity matrix $\mathbf{S}$ is a non-negative matrix and the transport plan is enforced to non-negative.

2. Monotonicity. Consider a subset $\mathcal{P}_1$ and define a set $\mathcal{P}_2 = \mathcal{P}_1 \cup \{\mathbf{x}_i\}$ for any $\mathbf{x}_i \notin \mathcal{P}_1$. To prove monotonicity of $h(.)$, it is sufficient to show that $h(\mathcal{P}_2) \geq h(\mathcal{P}_1)$. To this end, let $\boldsymbol{\gamma}_{\mathcal{P}_1}$ be the argmax for $h(\mathcal{P}_1)$ and consider the sub-matrix $\boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1}, :)$ which is the restriction of the optimal solution to the points in $\mathcal{P}_1$. We note that $\boldsymbol{\gamma}_{\mathcal{P}_1}(i,:) = \mathbf{0}; \mathbf{x}_i \notin \mathcal{P}_1$. We construct a feasible transport plan $\hat{\boldsymbol{\gamma}}$ for the set $\mathcal{P}_2$ as:

$$\hat{\boldsymbol{\gamma}}(\mathcal{I}_{\mathcal{P}_2}, :) = \left[\boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1}, :)^\top, \frac{\mathbf{1}}{n}\right]^\top,$$

and $\hat{\boldsymbol{\gamma}}(j,:) = \mathbf{0}$ for $\mathbf{x}_j \notin \mathcal{P}_2$. Let $\hat{h}(\mathcal{P}_2; \hat{\boldsymbol{\gamma}})$ indicate the function value evaluated at the feasible transport plan $\hat{\boldsymbol{\gamma}}$ for the set $\mathcal{P}_2$. We then have

$$\begin{aligned} h(\mathcal{P}_2) \geq \hat{h}(\mathcal{P}_2; \hat{\boldsymbol{\gamma}}) &= \langle \mathbf{S}(\mathcal{I}_{\mathcal{P}_2}, :), \hat{\boldsymbol{\gamma}}(\mathcal{I}_{\mathcal{P}_2}, :)\rangle \\ &= \langle \mathbf{S}(\mathcal{I}_{\mathcal{P}_1}, :), \boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1}, :)\rangle + \left\langle \mathbf{S}(i,:), \frac{\mathbf{1}}{n}\right\rangle \\ &= h(\mathcal{P}_1) + \left\langle \mathbf{S}(i,:), \frac{\mathbf{1}}{n}\right\rangle \\ &\geq h(\mathcal{P}_1) (\text{Since } \mathbf{S}(i,:) \geq \mathbf{0}.) \end{aligned} \tag{9}$$

3. Super-additivity over disjoint sets. Consider two disjoint sets $\mathcal{P}_1$ and $\mathcal{P}_2$. Let $\boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1},:)$ and $\boldsymbol{\gamma}_{\mathcal{P}_2}(\mathcal{I}_{\mathcal{P}_2},:)$ represent the sub-matrices of the respective optimal solutions to the points in $\mathcal{P}_1$ and $\mathcal{P}_2$. For the disjoint union set $\mathcal{P} = \mathcal{P}_1 \cup \mathcal{P}_2$, we construct a feasible transport plan $\hat{\boldsymbol{\gamma}}$ as:

$$\hat{\boldsymbol{\gamma}}(\mathcal{I}_{\mathcal{P}},:) = \left[\boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1},:)^\top, \boldsymbol{\gamma}_{\mathcal{P}_2}(\mathcal{I}_{\mathcal{P}_2},:)^\top\right]^\top,$$

and $\hat{\boldsymbol{\gamma}}(j,:) = \mathbf{0}$ for $\mathbf{x}_j \notin \mathcal{P}$. Evaluating the function $\hat{h}(\mathcal{P};\hat{\boldsymbol{\gamma}})$ at the feasible solution, we get

$$\begin{aligned} h(\mathcal{P}) \geq \hat{h}(\mathcal{P};\hat{\boldsymbol{\gamma}}) &= \langle \mathbf{S}(\mathcal{I}_{\mathcal{P}},:), \hat{\boldsymbol{\gamma}}(\mathcal{I}_{\mathcal{P}},:)\rangle \\ &= \langle \mathbf{S}(\mathcal{I}_{\mathcal{P}_1},:), \boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1},:)\rangle + \langle \mathbf{S}(\mathcal{I}_{\mathcal{P}_2},:), \boldsymbol{\gamma}_{\mathcal{P}_2}(\mathcal{I}_{\mathcal{P}_2},:)\rangle \\ &= h(\mathcal{P}_1) + h(\mathcal{P}_2). \end{aligned} \tag{10}$$

□

Lemma 2. *The optimization problem defined in (7) is a non-negative, monotone, submodular maximization problem subject to cardinality constraint k.*

Proof. We derive all the three properties below.

1. Non-negativity: $f(\mathcal{P}) \geq 0\ \forall \mathcal{P} \subseteq \mathcal{S}$. The proof follows along similar lines of the non-negativity proof in Lemma 1.

2. Monotonicity: $f(\mathcal{P}_2) \geq f(\mathcal{P}_1)\ \forall \mathcal{P}_1 \subseteq \mathcal{P}_2 \subseteq \mathcal{S}$. Akin to the monotonicity proof in Lemma 1, for a super-set $\mathcal{P}_2 = \mathcal{P}_1 \cup \{\mathbf{x}_i\}; \mathbf{x}_i \notin \mathcal{P}_1$, we can construct a feasible transport $\hat{\boldsymbol{\gamma}}$ using the optimal solution $\boldsymbol{\gamma}_{\mathcal{P}_1}$ as:

$$\hat{\boldsymbol{\gamma}}(\mathcal{I}_{\mathcal{P}_2},:) = \left[\boldsymbol{\gamma}_{\mathcal{P}_1}(\mathcal{I}_{\mathcal{P}_1},:)^\top, \frac{\mathbf{v}}{\|\mathbf{v}\|_1}\right]^\top,$$

where $\mathbf{v} = k\mathbf{1}/n - \boldsymbol{\gamma}_{\mathcal{P}_1}^\top \mathbf{1} \geq \mathbf{0}$. Following similar lines to the argument in Lemma 1, we obtain the monotonicity result.

3. Submodularity: To prove submodularity of function $f(\mathcal{P})$ in (7), we first note the following result (Kawano et al., 2022, Lemma 2).

Lemma 6. *[(Kawano et al., 2022, Lemma 2)] Let l, m, n be positive integers. Given a positive valued $m \times n$ matrix $\mathbf{S} > 0$, the following set function $\psi : 2^m \to \mathbb{R}_+$ is a submodular function:*

$$\psi(\mathcal{P}) = \max_{\boldsymbol{\gamma} \in \Gamma_{\leq}(\mathbf{1}_{\mathcal{P}}/l, \mathbf{1}_n/n)} \langle \mathbf{S}, \boldsymbol{\gamma}\rangle, \tag{11}$$

where as defined earlier, $\Gamma_{\leq}(\boldsymbol{\mu},\boldsymbol{\nu}) = \{\boldsymbol{\gamma} \in \mathbb{R}^{m\times n} \mid \boldsymbol{\gamma} \geq 0, \boldsymbol{\gamma}\mathbf{1} = \boldsymbol{\mu}, \boldsymbol{\gamma}^\top\mathbf{1} \leq \boldsymbol{\nu}\}$ and $\mathbf{1}_n$ is a $n \times 1$ vector of 1.

We observe that for $l = k$, (11) is equivalent to the proposed function $f(\cdot)$ defined in (7) as follows:

- For a given $\mathcal{P}$, let $\boldsymbol{\gamma}_1$ be an optimal solution for (11). Then, $\boldsymbol{\gamma}_2 = k\boldsymbol{\gamma}_1$ is an optimal coupling for computing $f(\mathcal{P})$ in (7). Similarly, if $\boldsymbol{\gamma}_2$ be an optimal solution for computing $f(\mathcal{P})$ in (7), then $\boldsymbol{\gamma}_1 = \boldsymbol{\gamma}_2/k$ is an optimal solution for computing $f(\mathcal{P})$ in (11).
- Hence, for a given $\mathcal{P}$, $f(\mathcal{P}) = k\psi(\mathcal{P})$

Due to the above, $\forall A, B \subseteq \mathcal{S}$

$$\psi(A \cup B) + \psi(A \cap B) \leq \psi(A) + \psi(B) \Rightarrow f(A \cup B) + f(A \cap B) \leq f(A) + f(B),$$

which proves that f is a submodular function.

□

Lemma 3. *Let $\mathcal{P}^*$ of cardinality k be an optimal solution of (6). Then $\mathcal{P}^*$ is also an optimal solution of (7), and vice-versa.*

Proof. Recall that for any set $\mathcal{P}$ of cardinality k, $f(\mathcal{P}) = h(\mathcal{P})$. Due the monotonicity properties in Lemmas 1 and 2, we can restrict the feasible region in problems (6) and (7) only across sets of cardinality k where they are equivalent, and have the same optimal solution. □

Lemma 4. *Let $\hat{\mathcal{P}}$ be the classical greedy solution of (7) with $|\hat{\mathcal{P}}| = k$. Let $\mathrm{OPT} = h(\mathcal{P}^*)$, where $\mathcal{P}^*$ is an optimal solution of (6). Then, $h(\hat{\mathcal{P}}) \geq (1 - 1/e)\mathrm{OPT}$.*

Proof. Recall that for any set $\mathcal{P}$ with $|\mathcal{P}| \leq k$, $f(\mathcal{P}) \geq h(\mathcal{P})$. As $|\hat{\mathcal{P}}| = k$, we have the equality $f\left(\hat{\mathcal{P}}\right) = h\left(\hat{\mathcal{P}}\right)$. Applying the classical greedy approximation theorem in (Nemhauser et al., 1978), we get $h\left(\hat{\mathcal{P}}\right) = f\left(\hat{\mathcal{P}}\right) \geq (1 - 1/e)\, f\left(\mathcal{P}^*\right) \geq (1 - 1/e)\, OPT$. □

Lemma 5. *Let $\alpha_{j,\min}$ denote $\frac{1}{\lfloor n/k \rfloor}$ times the sum of the $\lfloor n/k \rfloor$ smallest entries of the vector $\mathbf{S}(j,:)$, and let $\alpha_{j,\max}$ denote $\frac{1}{\lfloor n/k \rfloor}$ times the sum of the $\lfloor n/k \rfloor$ largest entries of $\mathbf{S}(j,:)$. Define $\alpha = \min_{j\in[m]} \frac{\alpha_{j,\min}}{\alpha_{j,\max}}$. Let $\hat{\mathcal{P}}$ be the solution returned by the greedy algorithm for (7), where the proposed approximate marginal gain function (8) is used in each iteration and $|\hat{\mathcal{P}}| = k$. Then, $f(\hat{\mathcal{P}}) = h(\hat{\mathcal{P}}) \geq (1 - e^{-\alpha})\,\mathrm{OPT}$, where $\mathrm{OPT} = f(\mathcal{P}^*) = h(\mathcal{P}^*)$ and $\mathcal{P}^*$ is an optimal solution to (7) with $|\mathcal{P}^*| = k$.*

Proof. At the iteration $i+1$, let $\mathbf{x}^* = \underset{\mathbf{x}\in\mathcal{S}\setminus\mathcal{P}_i}{\arg\max}\, \hat{f}(\mathbf{x}|\mathcal{P}_i)$ be the point that maximizes the approximate marginal gain function (8), which is used to update the solution to $\mathcal{P}_{i+1} = \mathcal{P}_i \cup \{\mathbf{x}^*\}$. Then,

$$f\left(\mathbf{x}^*|\mathcal{P}_i\right) \geq \hat{f}\left(\mathbf{x}^*|\mathcal{P}_i\right) \geq \frac{1}{k} \sum_{\mathbf{x}_l\in\mathcal{P}^*\setminus\mathcal{P}_i} \left[\hat{f}\left(\mathbf{x}_l|\mathcal{P}_i\right)\right] \tag{12}$$

Further, for any $\mathbf{x}_l \notin \mathcal{P}_i$ let $\mathcal{P} = \mathcal{P}_i \cup \{\mathbf{x}_l\}$. We derive the inequality

$$\begin{aligned} f\left(\mathbf{x}_l|\mathcal{P}_i\right) = f\left(\mathcal{P}\right) - f\left(\mathcal{P}_i\right) &= \left\langle \mathbf{S}\left(\mathcal{I}_{\mathcal{P}},:\right), \boldsymbol{\gamma}_{\mathcal{P}}\left(\mathcal{I}_{\mathcal{P}},:\right)\right\rangle - \left\langle \mathbf{S}\left(\mathcal{I}_{\mathcal{P}_i},:\right), \boldsymbol{\gamma}_{\mathcal{P}_i}\left(\mathcal{I}_{\mathcal{P}_i},:\right)\right\rangle \\ &\leq \left\langle \mathbf{S}\left(\mathcal{I}_{\mathcal{P}},:\right), \boldsymbol{\gamma}_{\mathcal{P}}\left(\mathcal{I}_{\mathcal{P}},:\right)\right\rangle - \left\langle \mathbf{S}\left(\mathcal{I}_{\mathcal{P}_i},:\right), \boldsymbol{\gamma}_{\mathcal{P}}\left(\mathcal{I}_{\mathcal{P}_i},:\right)\right\rangle \\ &= \left\langle \mathbf{S}(l,:), \boldsymbol{\gamma}_{\mathcal{P}}(l,:)\right\rangle \\ &\leq \alpha_{l,\max}. \end{aligned}$$

The inequality in the second line follows from the fact that $\boldsymbol{\gamma}_{\mathcal{P}_i}$ is the $\arg\max$ for the set $\mathcal{P}_i$ in (7) and $\boldsymbol{\gamma}_{\mathcal{P}}\left(\mathcal{I}_{\mathcal{P}_i},:\right)$ — appended with $\mathbf{0}$ for other rows— is one of the feasible solution. Likewise, $\hat{f}\left(\mathbf{x}_l|\mathcal{P}_i\right) \geq \alpha_{l,\min}$. Hence,

$$\hat{f}\left(\mathbf{x}_l|\mathcal{P}_i\right) \geq \frac{\alpha_{l,\min}}{\alpha_{l,\max}} f\left(\mathbf{x}_l|\mathcal{P}_i\right). \tag{13}$$

Pugging the inequality (13) in (12) we have

$$f\left(\mathbf{x}^*|\mathcal{P}_i\right) \geq \frac{1}{k} \sum_{\mathbf{x}_l\in\mathcal{P}^*\setminus\mathcal{P}_i} \left[\frac{\alpha_{l,\min}}{\alpha_{l,\max}} f\left(\mathbf{x}_l|\mathcal{P}_i\right)\right] \geq \frac{\alpha}{k} \sum_{\mathbf{x}_l\in\mathcal{P}^*\setminus\mathcal{P}_i} \left[f\left(\mathbf{x}_l|\mathcal{P}_i\right)\right]. \tag{14}$$

Leveraging the submodular and monotonic properties of the function $f(\cdot)$ from Lemma 2, we obtain the inequalities

$$\sum_{\mathbf{x}_l\in\mathcal{P}^*\setminus\mathcal{P}_i} f\left(\mathbf{x}_l|\mathcal{P}_i\right) \geq f\left(\mathcal{P}_i \cup \left(\mathcal{P}^* \setminus \mathcal{P}_i\right)\right) - f\left(\mathcal{P}_i\right) \geq f\left(\mathcal{P}^*\right) - f\left(\mathcal{P}_i\right). \tag{15}$$

Noting that $f\left(\mathbf{x}^*|\mathcal{P}_i\right) = \left[f\left(\mathcal{P}^*\right) - f\left(\mathcal{P}_i\right)\right] - \left[f\left(\mathcal{P}^*\right) - f\left(\mathcal{P}_{i+1}\right)\right]$, and using (15) in (14) gives the recurrence relation

$$f\left(\mathcal{P}^*\right) - f\left(\mathcal{P}_{i+1}\right) \leq \left(1 - \frac{\alpha}{k}\right)\left[f\left(\mathcal{P}^*\right) - f\left(\mathcal{P}_i\right)\right],$$

frow which it follows that

$$f\left(\mathcal{P}^*\right) - f\left(\hat{\mathcal{P}}\right) \leq \left(1 - \frac{\alpha}{k}\right)^k \left[f\left(\mathcal{P}^*\right) - f\left(\varnothing\right)\right].$$

As $f\left(\varnothing\right) = 0$, we get the desired result namely,

$$f\left(\hat{\mathcal{P}}\right) \geq \left(1 - \left(1 - \frac{\alpha}{k}\right)^k\right) f\left(\mathcal{P}^*\right) \geq \left(1 - e^{-\alpha}\right)\mathrm{OPT}.$$

□

C IMPLEMENTATION DETAILS

C.1 Hardware and License

All models are implemented in `Python 3.10` using `PyTorch 2.3.0`. Image and language training are performed on servers with Intel(R) Xeon(R) Gold 6226R CPUs (2.90GHz) and three NVIDIA RTX A6000 GPUs. For language model pretraining, we use `JAX` (Bradbury et al., 2018) (v0.7.2) on the same GPU infrastructure.

C.2 Algorithm Implementation

Implementation of POT Objective: To solve the entropic-regularized partial optimal transport (OT) problem, we rely on the Python Optimal Transport (POT) library[2] . Specifically, we use the function `ot.partial.entropicwasserstein`[3],

which implements the entropic-regularized variant of partial OT. This formulation allows for transporting only a fraction of the total mass between the source and target distributions along with the enforcement of inequality on the marginals.

In our implementation, the cost matrix C is constructed using pairwise distances between features of the source and target prototypes, which could be Euclidean or cosine distances depending on the application. The fraction of transported mass τ and the entropic regularization λ are treated as hyperparameters. The function `ot.partial.entropic_wasserstein` efficiently returns both the optimal transport plan and the associated partial OT cost, which we use as the objective function $f(\cdot)$ in downstream optimization or prototype selection procedures.

A typical usage in Python is as follows:

```
import ot

# mu: source weights
# nu: target weights
# C: cost matrix
# tau: fraction of transported mass
# lambda: entropy regularization
T, cost = ot.partial.entropic_wasserstein(
    mu, nu, C,
    tau=tau,
    reg=lambda
)
```

The default maximum iterations parameter for this function is set adaptively along with a stopping threshold of $1e-6$.

Table 4: Source Size vs Maximum Iterations.

Source Set Size	Max Iterations
64-200	100
500-1000	1000
1000-4000	2000
5000	4000

C.3 Finetuning experiments

We adapt the codebase of (Nguyen et al., 2025) for all our finetuning experiments. We use Adam (Adam et al., 2014) with learning rate of 1e-5, gradient accumulation steps of 64 with batch size 1. We directly use raw LoRA gradients for constructing similarity matrices for CoLM, GREATS, `UniPROT`. For GREATS (Wang et al., 2024) we randomly sample few random points from train-set as anchors at every train step. For SBERT (Reimers and Gurevych, 2019), we use BERT-BASE-UNCASED as the embedding model, and construct similarity matrix from the embeddings instead of gradients.

C.4 Details of baselines

GREATS (Wang et al., 2024). GREATS formulates online batch selection as optimizing a set utility that measures the single-step reduction in validation loss under a gradient-descent update. Let w_t be the current parameters, B_t a candidate

[2] https://pythonot.github.io/.

[3] https://pythonot.github.io/gen_modules/ot.partial.html#ot.partial.entropic_partial_wasserstein.

batch, and $S \subseteq B_t$ a subset of size k. The ideal utility at iteration t is

$$\mathcal{U}^{(t)}(S; \mathbf{z}^{(\mathrm{val})}) \;:=\; \ell(\boldsymbol{\theta}_t, \mathbf{z}^{(\mathrm{val})}) \;-\; \ell\Big(\boldsymbol{\theta}_t - \eta_t \sum_{\mathbf{z}\in S} \nabla \ell(\boldsymbol{\theta}_t, \mathbf{z}),\, \mathbf{z}^{(\mathrm{val})}\Big),$$

and selection solves $\arg\max_{S\subseteq B_t, |S|=k} \mathcal{U}^{(t)}(S; \mathbf{z}^{(\mathrm{val})})$. Since exact evaluation is intractable, GREATS applies a lower-order Taylor approximation of the validation loss around $\boldsymbol{\theta}_t$ to obtain a closed-form surrogate for the marginal gain of adding a training point $\mathbf{z}$:

$$U^{(t)}(\mathbf{z} \mid S) \;\approx\; \eta_t\, \mathbf{g}(\mathbf{z})^\top \mathbf{g}\big(\mathbf{z}^{(\mathrm{val})}\big) \;-\; \eta_t^2\, \mathbf{g}(\mathbf{z})^\top H\big(\mathbf{z}^{(\mathrm{val})}\big)\, \mathbf{g}(\mathbf{z}^*),$$

where $\mathbf{g}(\cdot) = \nabla \ell(\boldsymbol{\theta}_t, \cdot)$, $\boldsymbol{\mathcal{H}}(\cdot)$ is the Hessian of the validation loss, and z^* denotes the current aggregate. In practice, $\boldsymbol{\mathcal{H}}$ is approximated (e.g., $\boldsymbol{\mathcal{H}} \approx I$), yielding a gradient inner-product scoring with a correction term. A greedy procedure iteratively adds the point with largest approximate marginal gain until k points are selected. To avoid materializing per-example model-sized gradients, GREATS computes all required gradient inner-products in a single backpropagation via a "ghost inner-product" reparameterization that expresses layerwise gradient inner-products using already-available activations and output gradients, and merges selection with the update without extra passes.

CoLM (Nguyen et al., 2025). CoLM casts mini-batch construction as coreset selection in gradient space for memory-efficient fine-tuning. CoLM first addresses imbalance by including *all* examples from "small" sources (those with insufficient sample count in the large batch), while selecting representatives (medoids) from each "big" source. To align selection with Adam, per-example gradients are normalized by the optimizer's exponential-moving-average statistics, yielding normalized directions proportional to $\boldsymbol{m}_t/(\epsilon + \sqrt{\boldsymbol{v}_t})$. To reduce dimensionality and denoise, CoLM estimates the gradient of the *last V-projection* parameters (e.g., LoRA V) using a zeroth-order SPSA estimator with two perturbed forward passes and precached penultimate activations, then sparsifies by keeping the coordinates with largest normalized magnitudes. Within each big source, a greedy medoid selection is performed in the projected, sparsified, Adam-normalized gradient space so that the aggregated coreset gradient approximates that of the full large batch; the final mini-batch is the union of all small-source examples and the selected big-source medoids.

SBERT (Reimers and Gurevych, 2019). SBERT modifies BERT into siamese/triplet architectures with shared weights that encode each sentence independently. A fixed-size embedding $u \in \mathbb{R}^d$ is obtained via a pooling operation over token representations (commonly mean pooling). Training uses sentence-pair supervision: (i) a classification objective on NLI pairs, where a classifier consumes a concatenation of functions of the two embeddings (e.g., $[u; v; |u - v|]$) to predict the label; (ii) a regression objective for semantic textual similarity, where the cosine of (u, v) is regressed to a gold score via MSE; and (iii) optionally, triplet loss $\max\{0, \cos(u_a, u_n) - \cos(u_a, u_p) + \gamma\}$ for anchor–positive–negative tuples. At inference, sentence embeddings are compared with cosine or dot-product for retrieval and clustering.

GradNorm (Katharopoulos and Fleuret, 2018). Given a large batch $\mathcal{B}$, compute per-example gradient features and rank by norm. For parameters θ and loss $\ell_i = \ell(f_{\boldsymbol{\theta}}(\mathbf{x}_i), y_i)$, define raw gradient $\boldsymbol{g}_i = \nabla_{\boldsymbol{\theta}} \ell_i$. To align with Adam, each coordinate is normalized as

$$\tilde{\boldsymbol{g}}_i = \frac{\boldsymbol{m}_t}{\sqrt{\boldsymbol{v}_t} + \epsilon} \odot \boldsymbol{g}_i$$

where $(\boldsymbol{m}_t, \boldsymbol{v}_t)$ are the exponential moving averages of first and second moments. Instead of ℓ_2 distance, GradNorm computes similarity in this normalized gradient space using cosine:

$$s_{ij} \;=\; \frac{\langle \tilde{\boldsymbol{g}}_i, \tilde{\boldsymbol{g}}_j \rangle}{\|\tilde{\boldsymbol{g}}_i\|_2 \, \|\tilde{\boldsymbol{g}}_j\|_2}.$$

Each example is scored by its (smoothed) gradient norm $\|\tilde{\boldsymbol{g}}_i\|_2$, and the top-$k$ are selected. This yields a subset whose update direction emphasizes examples with largest effective gradient magnitude under the optimizer's scaling.

MaxLoss (Shalev-Shwartz and Wexler, 2016). Each example $i \in \mathcal{B}$ is scored by its instantaneous loss $s_i = \ell(f_{\boldsymbol{\theta}}(x_i), y_i)$ The k highest-loss items are selected to form the training subset. This "hard-example" criterion requires only forward passes and captures points where the current model performs worst. Optionally, per-example losses can be combined with Adam-smoothed gradient norms to provide importance weights during optimization.

C.5 Calculation of gradient features

Let $\mathbf{z}_i$ denote an example, $\ell(\boldsymbol{\theta}; \mathbf{z}_i)$ the training loss, and let $\mathbf{W}^{(L)}_{V,\text{LoRA}}$ be the parameter tensor of the *last* transformer block's value projection adapted by LoRA.[4] We flatten $\mathbf{W}^{(L)}_{V,\text{LoRA}}$ to a vector $v \in \mathbb{R}^{d_{\text{vp}}}$. At iteration t and current parameters $\boldsymbol{\theta}_t$, we compute per-example gradients as

$$\mathbf{g}^{\text{vp}}_{i,t} \;:=\; \nabla_v \, \ell(\boldsymbol{\theta}_t; \mathbf{z}_i) \;\in\; \mathbb{R}^{d_{\text{vp}}},$$

where the gradients are restricted to the LoRA-adapted last V-projection. Unlike the zeroth-order MeZO estimator used in (Nguyen et al., 2025), these gradients are obtained directly by backpropagation.

Adam-aligned normalization. To align with the update rule of Adam, we normalize each per-example gradient using the optimizer's moment statistics. Let $m_t, v_t \in \mathbb{R}^{d_{\text{vp}}}$ denote the first and second moment accumulators,

$$\boldsymbol{m}_t \;=\; \beta_1 \boldsymbol{m}_{t-1} + (1-\beta_1)\, \bar{\mathbf{g}}_t, \qquad v_t \;=\; \beta_2 v_{t-1} + (1-\beta_2)\, \bar{\mathbf{g}}_t^2, \tag{16}$$

with $\beta_1, \beta_2 \in (0,1)$, $\epsilon > 0$, and $\bar{\mathbf{g}}_t$ the average gradient over the current pool. The normalized gradient feature for an example $\mathbf{z}_i$ is then

$$\boldsymbol{\phi}_{i,t} \;:=\; \frac{\boldsymbol{g}^{\text{vp}}_{i,t}}{\epsilon + \sqrt{\boldsymbol{v}_t}} \;\in\; \mathbb{R}^{d_{\text{vp}}}, \tag{17}$$

where the division is elementwise. These features are stored and subsequently used in similarity computations.

C.6 Pretraining Experiments

We implement the pretraining setup using `JAX` (Bradbury et al., 2018), primarily due to its just-in-time (JIT) compilation framework and empirical 2–3× training speedups over `PyTorch`. For the base architecture, we pretrain a LLAMA-3 model consisting of approximately 500M parameters on the OpenWebText corpus[5] (Radford et al., 2019). The training is carried out for 20k training steps with an effective batch size of 64 sequences, each of length 512. Optimization is performed using the Adam algorithm with a fixed learning rate of 1×10^{-4}, without auxiliary learning rate schedules.

For all methods involving prototype-based subset selection, we begin with a candidate batch of 128 sequences and select 50% prototypes, resulting in an effective batch size of 64 sequences used for parameter updates. Subset selection is performed on a per-batch basis, without leveraging history across iterations.

In the case of `UniPROT`, the underlying optimal transport (OT) problem is solved using 20 Sinkhorn iterations with entropic regularization strength set to 1×10^{-2}. The similarity (cost) matrices are constructed directly from the last-layer gradients of the model, and no additional low-pass filtering, smoothing, or adaptive reweighting is applied. Throughout, cosine similarity is used as the base kernel to define pairwise affinities.

The dataset is partitioned into a 95% training split and a 5% validation split.

D EXPERIMENTAL SETUP DETAILS

D.1 Model Details

PHI-2 (2.7B). PHI-2 is a 2.7B parameter model trained with an emphasis on mathematical and logical reasoning, derived from curated synthetic corpora and filtered web data. It supports a context length of 2,048 tokens. In our fine-tuning setup, we apply LoRA adapters (rank 128, $\alpha = 512$, dropout 0.05) to all attention projection matrices (QKV) and the two feed-forward layers.

PHI-3 family. We experiment primarily with the 3.8B variant (PHI-3 MINI), though the broader family also includes 7B and 14B models. The PHI-3 series continues the focus on compact models optimized for reasoning tasks, with available context lengths of 4K and 128K tokens depending on variant. Similar to PHI-2, we apply LoRA adapters to QKV projections and feed-forward layers during fine-tuning.

STABLELM ZEPHYR 3B. STABLELM ZEPHYR 3B is a 3B parameter instruction-tuned model designed as a general-purpose assistant, without a specific emphasis on mathematical reasoning. It supports input sequences up to 4K tokens. For LoRA fine-tuning, we insert adapters into all attention projection matrices (QKVO).

[4] "Last" refers to the topmost transformer block in the forward stack.

[5] https://huggingface.co/datasets/Skylion007/openwebtext

D.2 Datasets

For image settings we do on MNIST, CIFAR10, CIFAR100 as well as synthetic distributions.

For the mathematical reasoning experiments, we fine-tune on the **MATHINSTRUCT** dataset (Yue et al., 2023), which contains roughly 260K instruction–response pairs. The data is aggregated from 14 open-source mathematics corpora, covering diverse subfields and spanning a broad range of difficulty levels. The composition of MathInstruct is highly imbalanced—the largest constituent source is nearly 300 times larger than the smallest—and the detailed distribution across sources is provided in Figure 4a of the Appendix. Prior work has shown that fine-tuning on MathInstruct leads to state-of-the-art results on multiple standardized mathematical reasoning benchmarks.

For classification experiments, we additionally use three datasets from the SUPERGLUE benchmark (Wang et al., 2019): SST-2, CB, and MultiRC. For CB, we retain the complete training set of 250 labeled examples. For SST-2 and MultiRC, we randomly subsample 3K examples each for training.

D.3 Training Details

Following the configuration in Yue et al. (2023), we employ a learning rate of 2×10^{-5} with a cosine decay scheduler. The learning rate is linearly warmed up from 0 to 2×10^{-5} during the first 3% of training steps and subsequently decays to 0 following a cosine schedule. We fix the maximum sequence length to 512 tokens. Unless otherwise stated, all experiments on MATHINSTRUCT are trained for the equivalent of 1K gradient update steps. To enable larger effective batch sizes, we use gradient accumulation with an accumulation factor of 8.

For parameter-efficient fine-tuning, we adopt LoRA (Hu et al., 2022) with rank 128, scaling parameter $\alpha = 512$, and a dropout rate of 0.05. On PHI models, LoRA adapters are applied to all attention projection matrices (QKV) as well as the two feed-forward layers. On ZEPHYR, we apply LoRA to all attention projections (QKVO). All experiments are conducted on 3 NVIDIA A6000 GPUs, and each configuration is repeated three times to account for variance in training.

D.4 Evaluation Datasets and Metrics

Following Yue et al. (2023), we evaluate our models on a diverse suite of mathematical reasoning benchmarks spanning both in-domain and out-of-domain distributions.

In-domain benchmarks. The in-domain evaluation covers three widely used datasets: GSM8K (), MATH (**?**), and NUMGLUE (Mishra et al., 2022). GSM8K focuses on grade-school arithmetic word problems, MATH contains high-school competition-style problems across 29 mathematical domains, and NUMGLUE extends natural language understanding tasks with quantitative reasoning components.

Out-of-domain benchmarks. To test generalization beyond the training distribution, we additionally include SVAMP, the MATHEMATICS dataset, and SIMULEQ. These datasets emphasize robustness across algebraic manipulations, probability and statistics, number theory, and systems of equations, while also incorporating instances requiring multi-step logical reasoning and commonsense knowledge.

D.5 Evaluation Setup

1NN Classifier Setup Let $\mathcal{X}$ and $\mathcal{Y}$ denote the source and target datasets, respectively, with potentially differing class distributions, and let $\mathcal{P} \subseteq \mathcal{X}$ be a candidate representative set for the target dataset $\mathcal{Y}$. The quality of $\mathcal{P}$ is assessed using a 1-nearest neighbour (1-NN) classifier parameterized by the elements of $\mathcal{P}$. Each instance $y \in Y$ is assigned the label of its nearest prototype in $\mathcal{P}$, where the ground-truth class labels of the elements in $\mathcal{P}$ are assumed to be available during this evaluation. The resulting classification accuracy serves as the evaluation metric for comparing prototype selection algorithms.

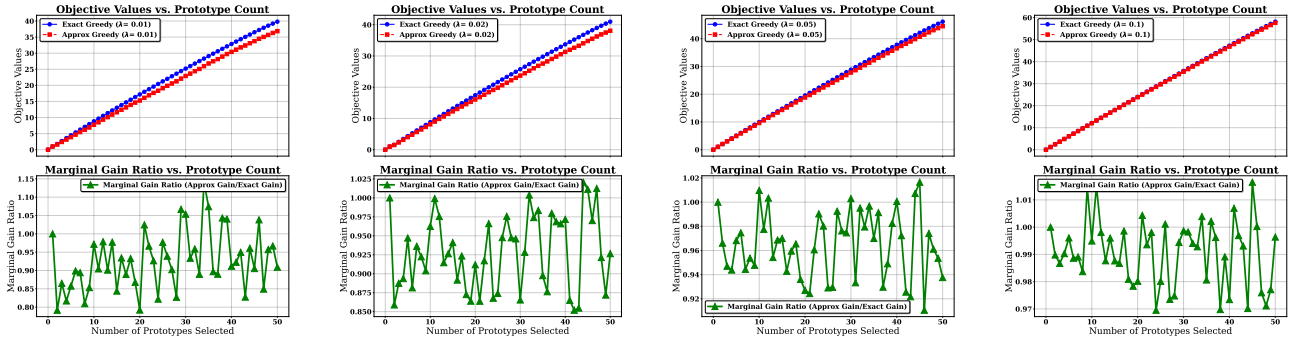
LLM-Finetuning: All questions are posed in an open-ended format. We adopt the standard *exact match* metric, where a prediction is considered correct only if it exactly matches the gold reference solution. Evaluation is conducted under the 0-shot setting with a maximum decoding context length of 2048 tokens. We use the Program-of-Thought (PoT) prompting strategy as the default, and fall back to Chain-of-Thought (CoT) prompting when PoT is not applicable, following Yue et al. (2023).

E ADDITIONAL EXPERIMENTAL RESULTS

E.1 Additional Ablations on Entropic Regularization

To examine the influence of the entropic regularization parameter on the approximation error associated with the computation of the approximate marginal gain in Eq. (8), we conduct an ablation study under a controlled synthetic setting

derived from the CIFAR dataset. Specifically, we uniformly sample 2000 training instances from CIFAR and treat them identically as both the source and target sets, denoted by $\mathcal{S}$ and $\mathcal{T}$, respectively. As illustrated in Fig. 7, increasing the value of λ progressively reduces the approximation error toward zero, with the ratio between the approximate and exact marginal gains converging to unity and exhibiting low variance.



(a) Entropic Reg. ($\lambda = 0.01$) (b) Entropic Reg. ($\lambda = 0.02$) (c) Entropic Reg. ($\lambda = 0.05$) (d) Entropic Reg. ($\lambda = 0.1$)

Figure 7: Approximate vs. actual marginal gain as a function of the entropic regularization parameter λ. Increasing λ reduces the approximation error, with the marginal gain ratio converging to 1 and exhibiting low variance for larger λ values.

E.2 Additional Results on UniPROT-PB

Experiments on batch size=256 for UniPROT-PB

We experiment with UniPROT-PB batch size of 256 for selection instead of source-wise prototype selection. We fine-tune PHI-3 for 2048 steps (selection batch size 256) with prototype percentage 0.25, resulting in an effective batch size of 32. Table 5 shows that UniPROT continues to be effective even in the full-batch setting.

We compare against GREATS, GradNorm, and CoLM under the same training budget and report validation log-perplexity trajectories over optimization steps. Figure 8 shows that CoLM's validation perplexity degrades as batch size increases, while UniPROT-PB remains stable and continues to improve perplexity throughout training.

We report additional results on UniPROT-PB on **MATHINSTRUCT** dataset.

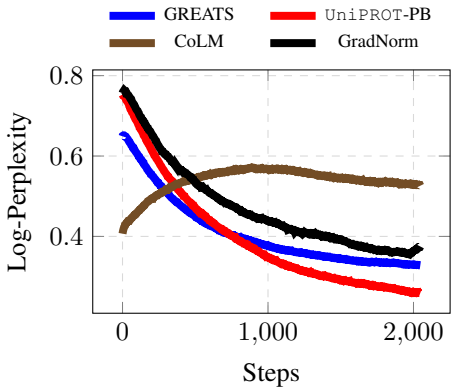


Figure 8: Validation perplexity when batch size=256 and prototype ratio is 25%.

Experiments on full-batch prototype selection

For this variant, we experiment full-batch selection instead of source wise prototype selection. We finetune PHI-3 for 2048 steps, selection batch size of 128, with prototype percentage as 0.5, resulting in a effective batch of 64. Table 5 shows that UniPROTcontinues to be effective even in full batch setting.

Table 5: Comparison of UniPROT-PB performance for $|\mathcal{B}| = 256$, following Yue et al. (2023) on **MATHINSTRUCT** dataset.

Method	Avg	In-domain			Out-of-domain		
		GSM8K	MATH	NumGLUE	SVAMP	Mathematics	SimulEq
COLM (Nguyen et al., 2025)	59.86	74.13	36.70	63.14	86.50	36.40	62.30
GREATS (Wang et al., 2024)	60.79	78.62	37.90	63.90	85.50	36.90	61.90
UniPROT-PB (Ours)	**61.34**	78.20	37.60	66.03	84.90	37.70	63.60

E.3 Additional Results on Zephyr-3B

Here, we report additional results on ZEPHYR-3B on MATHINSTRUCT Dataset.

Table 6: Per-batch performance across in-domain and out-of-domain datasets for PHI-3 on **MATHINSTRUCT**, batch size $|\mathcal{B}| = 128$ and total budget $k = 64$. We compare FT, GradNorm, COLM, GREATS, and UniPROT (Ours).

Method	In-domain				Out-of-domain				Avg-All
	GSM8K	MATH	NumGLUE	Avg	SVAMP	Mathematics	SimulEq	Avg	
FT (bs=64)	76.72	36.54	62.57	58.61	85.10	33.30	62.78	60.39	59.50
GradNorm	75.40	35.03	64.10	58.18	84.17	36.50	65.70	62.12	60.15
COLM	76.36	36.42	64.10	58.96	85.30	37.40	63.60	62.10	60.53
GREATS	77.80	37.28	64.40	59.83	85.00	38.00	62.06	61.69	60.76
UniPROT (Ours)	78.16	37.76	66.02	**60.65**	85.70	37.20	68.28	**63.73**	**62.19**

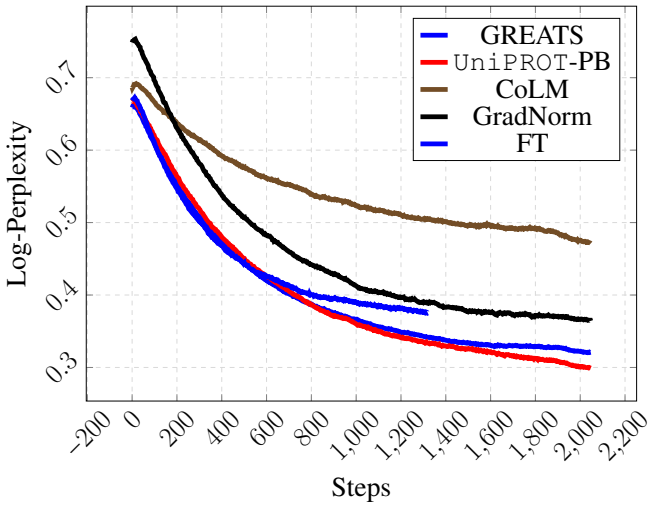


Table 7: Top: Per-batch performance of FT, GradNorm, COLM, GREATS, and UniPROT on MATHINSTRUCT with $|\mathcal{B}| = 128$, $k = 64$. Bottom: Validation perplexity when $|\mathcal{B}| = 128$, subset ratio 50%, and prototype selection is batch-wise.

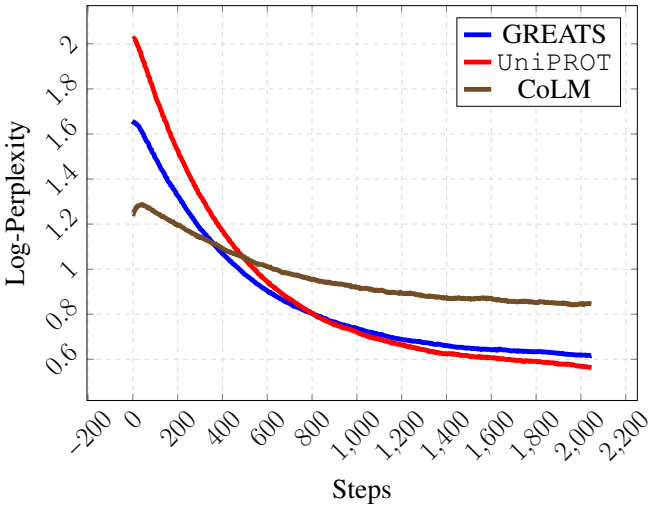


Figure 9: Zephyr-3b Validation perplexity when bs=128 and subset ratio 25% and prototype selection is batch wise.

Table 8: Performance across in-domain and out-of-domain datasets for ZEPHYR-3B on **MATHINSTRUCT**, batch size $|\mathcal{B}| = 128$ and total budget $k = 32$. Results are reported under two configurations for **all** baselines: *source-wise* (left of "/") and *batch-wise* (right of "/").

Method	In-domain				Out-of-domain				Avg-All
	GSM8K	MATH	NumGLUE	Avg	SVAMP	Mathematics	SimulEq	Avg	
FT	52.08	16.14	40.10	36.11	54.70	15.60	22.20	30.83	33.47
GradNorm	52.3 / 49.8	16.5 / 15.7	40.0 / 39.9	35.70	53.5 / 53.1	15.5 / 15.4	22.5 / 22.7	30.45	33.07
SBERT	51.3 / 21.7	14.6 / 14.01	41.6 / 27.44	28.44	56.3 / 34.5	16.2 / 13.6	21.98 / 13.8	26.06	27.25
COLM	50.6 / 49.88	**21.42** / 17.4	40.15 / 39.9	36.56	**55.8** / **54.1**	16.7 / 15.4	22.17 / 21.01	30.86	33.71
GREATS	52.8 / 50.6	19.01 / 15.9	40.8 / 38.8	36.32	54.5 / **54.6**	**17.1** / 13.9	21.98 / 21.78	30.64	33.48
UniPROT (Ours)	**54.4** / **54.89**	20.4 / **19.6**	**41.3** / **40.4**	**38.50**	54.3 / 54.1	16.5 / **15.5**	**24.7** / **22.95**	**31.34**	**34.92**

E.4 Additional Results on PHI-2

Here, we report additional results on PHI-2 on MATHINSTRUCT Dataset.

Table 9: Performance across in-domain and out-of-domain datasets for PHI-2 on **MATHINSTRUCT**, batch size $|\mathcal{B}| = 128$ and total budget $k = 32$. Results are reported under two configurations for **all** baselines: *source-wise* (left of "/") and *batch-wise* (right of "/").

Method	In-domain				Out-of-domain				Avg-All
	GSM8K	MATH	NumGLUE	Avg	SVAMP	Mathematics	SimulEq	Avg	
FT	59.28	14.80	47.88	40.65	**62.80**	19.40	36.70	39.63	40.14
GradNorm	58.6 / 60.3	15.1 / 14.6	51.3 / 50.6	41.75	61 / 59	20.2 / **20.4**	40.6 / 38.13	39.89	40.82
SBERT	58.4 / 45.5	12.43 / 9.83	52.01 / 48.4	37.76	61 / **64.2**	18.1 / 16.6	36.5 / 33.8	38.37	38.06
COLM	57.01 / 58.8	14.86 / 14.6	51.05 / 50.8	41.19	60 / 63.9	18.9 / 19.8	32.4 / 31.9	37.82	39.50
GREATS	59.5 / 58.4	15.4 / 15.6	51.2 / **54.1**	42.37	**61.3** / 61.5	**21.1** / 20.7	37.35 / 35.01	39.49	40.93
UniPROT (Ours)	**60.6** / 58.8	**16.8** / **16.7**	**51.61** / 51.7	**42.70**	61 / 62	20.9 / 19.8	**41.4** / **35.6**	**40.12**	**41.41**

Table 10: Comparison of PHI-2 **per-source** selection performance across in-domain and out-of-domain datasets, following Yue et al. (2023) on **MATHINSTRUCT** dataset, batch size $|\mathcal{B}| = 256$ and total budget $k = 32$.

Method	Avg	In-domain			Out-of-domain		
		GSM8K	MATH	NumGLUE	SVAMP	Mathematics	SimulEq
COLM-PS (Nguyen et al., 2025)	43.14	61.03	26.67	52.65	60.45	21.04	37.00
GREATS-PS (Wang et al., 2024)	43.73	61.92	27.21	52.40	62.00	20.80	38.03
UniPROT-PS (Ours)	**44.66**	62.40	27.63	53.97	64.72	20.30	38.91

F ADDITIONAL RELATED WORKS

OT for representation matching without selection. Wang et al. (2025) employ optimal transport (OT) to align class activation map (CAM) clusters with corresponding class prototypes. However, their use of OT is limited to *distribution matching* between pixel-level feature distributions and pre-defined cluster prototypes, rather than for subset or prototype selection. In particular, the prototypes are constructed as averages of pixel features assigned to each cluster (cf. Eq. (2) in Wang et al., 2025), and are not selected from a candidate pool under any combinatorial or cardinality constraint. Thus, no subset selection or prototype selection problem is formulated or solved in their framework.

Zhang et al. (2024) also leverage OT to compute token-to-prototype assignments by matching representations to their corresponding ground-truth prototypes. Unlike subset selection approaches, their formulation retains the full set of prototypes during optimization and uses entropy-regularized OT to obtain *soft assignments*. As a result, their method does not impose any sparsity, selection, or cardinality constraints on the set of active prototypes. Consequently, OT serves purely as an assignment mechanism rather than a tool for selecting a representative subset.

OT in coreset selection and dataset condensation. OT has also been explored in coreset selection and dataset condensation settings, where the goal is to construct a compact dataset that approximates the full data distribution. For example, Wasserstein-based coreset construction and dataset distillation methods (Zhao et al., 2021; Nguyen et al., 2021; Liu et al., 2025) leverage OT distances to measure distributional fidelity between synthetic and real data. However, these approaches typically *learn synthetic samples* or optimize continuous representations, rather than selecting a subset from a given discrete pool. As a result, they differ fundamentally from subset selection problems with combinatorial constraints.

OT barycenters and prototype learning. Another related direction involves OT barycenters, where prototypes are obtained as Wasserstein means of distributions (Cuturi and Doucet, 2014; Agueh and Carlier, 2011). These methods compute representative prototypes in a continuous space by averaging distributions under OT geometry. While effective for summarization, they do not select prototypes from an existing dataset and thus bypass the combinatorial nature of subset selection. Similarly, OT-based clustering methods rely on iterative refinement of centroids rather than discrete selection from a candidate set (Ho et al., 2017).

In contrast to the above works, our formulation uses *partial optimal transport* POT *as a selection principle*, where the objective explicitly optimizes over a constrained subset of prototypes under uniform weighting and cardinality restrictions. This leads to a fundamentally different regime in which OT directly governs subset selection, rather than serving solely as a matching or assignment operator.

Distinction to (Hong et al., 2024) (Hong et al., 2024) study mini-batch selection for efficient training of deep networks by selecting a subset of points from a given batch that maximizes group-wise orthogonalized representativeness. This objective is fundamentally different from our optimal transport (OT) formulation, which seeks to minimize the discrepancy between a prototypical distribution and a target distribution. In particular, their method operates *within* a single batch and does not incorporate any notion of a separate target set. Consequently, it is not directly applicable to settings where one aims to select a subset $\mathcal{P} \subseteq \mathcal{S}$ (from a source set $\mathcal{S}$) such that it matches a target set $\mathcal{T}$.

Algorithmically, their DivBS procedure performs greedy selection by iteratively choosing samples that maximize projection onto the current residual. This results in a weighted expansion of the full gradient, where each selected sample is assigned an importance coefficient based on its marginal contribution. Crucially, these coefficients are *non-uniform*: early selections capture dominant components (e.g., the mean), while subsequent selections explain diminishing residuals. Although the final mini-batch is treated uniformly during backpropagation (via averaging), the selection mechanism itself is inherently non-uniform and does not enforce equal importance among selected samples.

Our approach (UniPROT). In contrast, our method explicitly incorporates a *uniform weight constraint* into the optimization objective. We select prototypes by solving a constrained OT problem that minimizes the transport cost to the target distribution under the requirement that each prototype carries equal mass. This leads to a fundamentally different selection principle that ensures equal-importance representation while aligning the selected subset with the target distribution.

G BROADER IMPACT

This work develops a principled framework for prototype selection that aims to improve fairness and robustness in settings with distributional imbalance. By explicitly enforcing uniform weighting, `UniPROT`can reduce systematic underrepresentation of minority classes, which has positive implications for equitable model performance across demographic or domain groups. At the same time, more efficient subset selection methods could also be leveraged to accelerate training of harmful or biased systems if applied without safeguards. We believe that open discussion of both the benefits and limitations of prototype selection methods is important to ensure they are deployed responsibly, and that continued transparency in this line of work will help maximize positive societal impact.

H CODE

We release our code on GitHub.